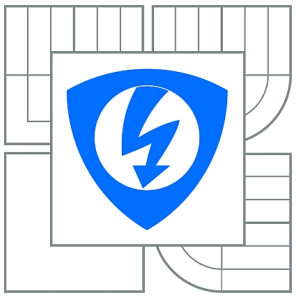
View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Digital library of Brno University of Technology

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ŠIFROVACÍ ALGORITMY LEHKÉ KRYPTOGRAFIE

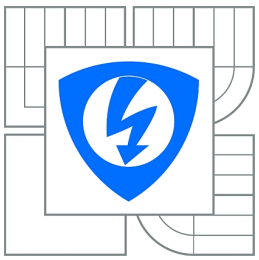
ENCODING ALGORITHMS OF SOFT CRYPTOGRAPHY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JIŘÍ HAVLÍČEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. VLASTIMIL ČLUPEK
SUPERVISOR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student: Jiří Havlíček ***ID:*** 140417
Ročník: 3 ***Akademický rok:*** 2012/2013

NÁZEV TÉMATU:

Šifrovací algoritmy lehké kryptografie

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a popište metody tzv. lehké kryptografie, určené pro využití kryptografických mechanizmů na méně výkonném HW. Popište možnosti implementace vybraných algoritmů lehké kryptografie. Vybrané algoritmy porovnejte a zhodnoťte.

DOPORUČENÁ LITERATURA:

[1] MENEZES, Alfred, VAN OORSCHOT, Paul, VANSTONE, Scott. Handbook of applied cryptography. Boca Raton : CRC Press, 1997. 780 s. ISBN 0849385237

[2] COLE, Peter, RANASINGHE, Damith. Networked RFID systems and lightweight cryptography: raising barriers to product counterfeiting. 2008. 355 s. ISBN 3540716408

Termín zadání: 11.2.2013 ***Termín odevzdání:*** 5.6.2013

Vedoucí práce: Ing. Vlastimil Člupek
Konzultanti bakalářské práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

Tato bakálářská práce se zabývá tzv. lehkou kryptografií určenou pro méně výkonný hardware.

V první části práce jsou vysvětleny základní pojmy a princip kryptografie, rozdíl mezi symetrickým a asymetrickým šifrováním s popisem nejrozšířenějších zástupců obou těchto odvětví kryptografie.

V práci jsou dále popsány a parametrově porovnány nově vyvinuté algoritmy se zaměřením na lehkou kryptografii. Pozornost je konkrétně věnována proudovým a blokovým šifrám a hashovacím funkcím. Následně se práce zabývá popisem hardwaru s omezeným výpočetním výkonem. Jedná se o zařízení omezené z hlediska napájení a velikosti samotných čipů. Popis je zaměřen na smart karty, RFID čipy a mikrokontroléry.

Praktická část je zaměřena na testování proudových šifer navržených pro softwarové implementace a na testování speciálních proudových šifer navržených pro lehkou kryptografii. Výsledky těchto testů poskytují ucelený pohled na výkonnostní rozdíly jednotlivých šifer určených pro rozdílné implementace.

KLÍČOVÁ SLOVA

Lehká kryptografie, proudové šifry, eSTREAM, hardwarově omezená zařízení

ABSTRACT

This bachelor's thesis deals with so called lightweight crptography which is specified for low-efficiency hardware.

The first part of my thesis explains basic terms and principles of cryptography, difference between symmetrical and asymmetrical encryption including description of the most widespread examples of both of these cryptography's branches.

The thesis continues with the description and parametric comparison of newly developed algorithms focusing on lightweight cryptography. I specially pay attention to current and block ciphers and hash functions. Afterwards the thesis describes limited computing power hardware. This is about device which is limited from the point of view of power supply and size of chips. The description is focused on smart cards, RFD chips and microcontrollers.

The practical part deals with testing of current ciphers which are designed for software implementations and with testing of current ciphers designed for lightweight cryptography. Results of the tests provide comprehensive view of differences of output of individual ciphers specified for different implementations.

KEYWORDS

Lightweight cryptography, stream ciphers, eSTREAM, hardware-limited devices

HAVLÍČEK, J. *Šifrovací algoritmy lehké kryptografie*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2013. 51 s. Vedoucí bakalářské práce Ing. Vlastimil Člupek.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Šifrovací algoritmy lehké kryptografie“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Vlastimilu Člupkovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

V úvodu první kapitoly je stručně popsána historie kryptografie až do poloviny 20. století, kdy dochází k rozvoji výpočetní techniky a začínáme mluvit o takzvané moderní kryptografii. Dále je zde vysvětleno rozdělení kryptologie na kryptografii a kryptoanalýzu, s popisem čím se obě vědy zabývají a jaký mají význam. Následně jsou vysvětleny základní pojmy užívané v kryptografii. V podkopitolách symetrické a asymetrické šifrování jsou s popisem výhod a nevýhod uvedeny nejpoužívanější zástupci těchto dvou podskupin kryptografie. Popis symetrických a asymetrických algoritmů je zaměřen na jejich princip, bezpečnost a na možnosti použití.

Druhá kapitola je věnována lehké kryptografii. V dnešní době vznikají zařízení s velmi malou plochou čipu a omezeným napájením. Do takovýchto zařízení je použití algoritmů klasické kryptografie popsané v první kapitole velmi obtížné, musí zde být implementovány algoritmy k tomuto účelu vyvinuté, tzv algoritmy lehké kryptografie. Zatím byly pro lehkou kryptografii vyvinuty blokové a proudové šifry a také hashovací funkce. Vybrané algoritmy tohoto odvětví kryptografie jsou v této kapitole popsány a parametrově porovnány.

V další kapitole je popsána softwarová a hardwarová implementace. Se zaměřením na důležitost správné implementace. Porovnány jsou obě implementace z pohledu bezpečnosti dat a jsou zde uvedeny výhody a nevýhody použití dané implementace. Další část této kapitoly je zaměřena na zařízení s omezeným výpočetním výkonem, mezi které řadíme smart karty, RFID čipy a mikrokontroléry. Pozornost je dále věnována obecnému popisu útoků na hardwarové implementace pomocí postranních kánalů.

Praktická část této práce je zaměřena na testování proudových šifer na stolním PC podle čtyř rozdílných kritérií. Testovány byly proudové šifry vhodné k softwarové implementaci a proudové šifry navržené pro hardwarově omezená zařízení. K testování byl využit volně přístupný testovací nástroj proudových šifer vytvořený projektem eSTREAM, testovány tedy byly proudové šifry zapojené do tohoto projektu. U vybraných šifer byl proveden test výkonu při šifrování dlouhých proudů dat, test rychlosti paketového šifrování, test agility šifer a test nastavení rychlosti klíče a inicializačního vektoru (IV). Výsledky testů jsou přehledně zpracovány a poskytují vypovídající informace o výkonu jednotlivých šifer. Výsledky výkonových rozdílů testovaných šifer, určených pro rozdílné implementace, jsou následně diskutovány.

1 KRYPTOGRAFIE

1.1 Historie

Již v dávných dobách byla snaha jakýmkoli způsobem utajit důležité zprávy. Komunikace utajená pomocí ukrytí zprávy bez nějaké její změny se nazývá steganografie, podle řeckých slov steganos (schovaný) a graphein (psát). V různých částech světa se rozvinuly různé formy steganografie. Například číňané psali zprávy na jemné hedvábí, které pak zmačkali do malé kuličky a zalili voskem. Posel pak voskovou kuličku polkl. Patří sem i neviditelné inkousty, použití bambusové hole pro přenášení zpráv a další důmyslné techniky [33].

Steganografie jasně ukazuje, že jde o techniku, jež sice poskytuje určitý stupeň utajení, má však zásadní nevýhodu. Když už se zprávu jednou podaří objevit, je prozrazena naráz. Pouhé její zachycení znamená ztrátu veškerého utajení [33].

Souběžně se steganografií se proto začala rozvíjet i kryptografie, jejíž název pochází z řeckého slova kryptos (skrytý). Cílem kryptografie není utajit existenci zprávy, ale její význam, a to pomocí šifrování. Aby nešlo zprávu přečíst, pozmění se podle pravidel předem dohodnutých mezi odesílatelem a příjemcem. Pokud taková zpráva padne do rukou nepříteli, je nečitelná. Nezná-li nepřítel použitá šifrovací pravidla, pak se mu podaří zjistit obsah zprávy jen s velkým úsilím, anebo vůbec ne [33].

Pro větší bezpečnost lze tyto nezávislé techniky kombinovat. Kryptografie je účinnější než steganografie, protože pomocí ní lze zabránit tomu, aby informace padla do rukou nepřítele, dokud nenalezne klíč k rozluštění zprávy [33].

Jednou z nejstarších kryptografických metod je takzvaná Ceaserova šifra. Její princip je založen na tom, že každé písmeno abecedy je během šifrování zaměněno za písmeno, které se v abecedě nachází o tři místa dále. Ceasar používal posun o tři místa, ale lze použít libovolné posunutí od 1 do 25. Jakékoli šifrování pomocí posunu písmen označujeme jako substituční šifrování. Ovšem postupem času se stalo toto šifrování velice slabím a bylo třeba použít důmyslnější techniky šifrování [27].

Další šifrou byla Vigenèrova šifra, která byla vynalezena v 16. století a zdála se neprolomitelná. Síla Vigenèrovy šifry spočívá v tom, že k zašifrování zprávy nepoužívá jednu, ale 26 odlišných šifrových abeced. Podstata spočíva v tom, že každé písmeno v textu je možné zakódovat pomocí jiné abecedy. Tato šifra byla nakonec prolomena v 19. století a přestala být tedy bezpečná [33].

Jediná šifra, která je ve svém principu prozatím nerozluštitelná je tzv. Vernamova šifra. Tato šifra spočívá ve sčítání znaků zprávy s heslem. Heslo je však složeno z jednorázově vygenerovaných náhodných znaků a je dlouhé stejně jako délka zprávy.

Není proto žádný způsob, jak zjistit vztahy mezi znaky a šifru rozluštit. Pro velkou náročnost se tato šifra používá pouze pro velice důležité zprávy [11].

Vzhledem k jednoduchosti některých šifer, které většinou není obtížné prolomit můžeme kombinací dvou nebo více šifer získat systém, který je o poznání silnější. Tomuto postupu se říka skládání šifer [27].

V první polovině 20. století vynalezl Arthur Scherbius náhradu za zastaralé šifrovací systémy, založené na tužce a papíru. Vyvinul šifrovací zařízení pojmenované Enigma. Šifrovací mechanismus použitý v Enigmě byl propracovaný a složitý, jeho základní verze měla 10^{20} možných klíčů, což je téměř tolik jako měli donedávna některé používané algoritmy. Z počátku se zdálo, že je tento systém neprolomitelný, ovšem jak je známo, podařilo se Enigmu prolomit a to především díky chybám při správě klíčů a při nesprávném použití [33].

S nástupem moderní počítačové techniky a internetu dostal pojem kryptografie zcela nový rozměr. Význam tohoto oboru roste neuvěřitelnou rychlostí tak, jak dochází k vývoji počítačové techniky a s tím spojeným rostoucím výpočetním výkonem. Ten je totiž v případě kryptografie spíše nepřítel, protože umožňuje tzv. útoky hrubou silou. To znamená vyzkoušení všech klíčů. S rostoucím výpočetním výkonem se zkracuje doba potřebná k rozluštění zašifrované zprávy [11].

Neustále jsou vyvíjeny lepší a silnější algoritmy, používající stále delší klíče. Dá se říci, že moderní kryptografie se vydává třemi různými cestami, z nichž každá je vhodná pro jiné použití. Jsou to symetrická kryptografie, asymetrická kryptografie a kvantová kryptografie, jež se využívá je pro bezpečný přenos klíčů [11].

1.2 Základní pojmy

Kryptologie je souhrnným označením pro kryptografii a kryptoanalýzu [27].

Kryptografie je věda, která s využitím matematiky šifruje a dešifruje data. Umožnuje uschovat citlivé informace nebo je prenášet přes nezabezpečené sítě jako je internet. V souvislosti s kryptografií je potřeba vysvětlit několik důležitých základních pojmů [13]:

- Šifra – šifrou rozumíme algoritmus, který pomocí matematických funkcí převádí otevřený text (nešifrovaná data) do zašifrované podoby.
- Klíč – jedná se o číselný parametr, který vstupuje do šifrovacího algoritmu, za jehož pomoci se převádí otevřený text do šifrované podoby a naopak zašifrovaný text do otevřené podoby.
- Symetrická šifra – je algoritmus využívající pro šifrování i dešifrování stejného klíče. Tento klíč musí být tajný.

- Asymetrická šifra – je algoritmus využívající dva různé klíče, jeden pro šifrování a druhý pro dešifrování. Jeden klíč je veřejný a druhý soukromý.
- Hašovací funkce – je to jednosměrná funkce, která převádí data o libovolné velikosti na posloupnost bitů o konstantní délce (hash kód).
- Digitální podpis – je zašifrovaný otisk z elektronického dokumentu, který jednoznačně identifikuje vlastníka tohoto dokumentu.

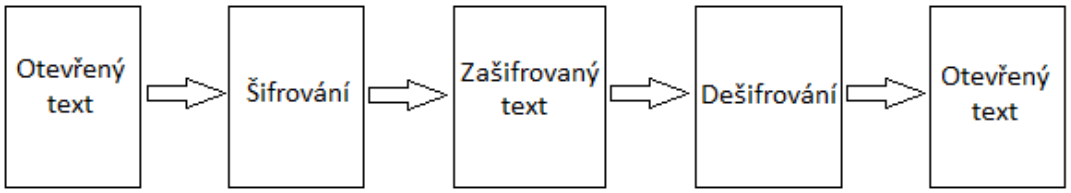
Kryptoanalýza se zabývá luštením šifer a textů. Dnes je známo mnoho kryptoanalytických metod, z nichž zde jsou ty nejběžnější [18]:

- Útok hrubou silou — je univerzální princip, kdy se útočník snaží vyzkoušet všechny možné vyrianty klíče. Z toho vyplývá, že čím máme delší klíč, tím je útok více náročný na čas a výpočetní výkon. Například vyzkoušení všech různých kombinací 128 bitového klíče by při rychlosti 10^6 MIPS trvalo 10^{28} let. Podle toho, jak chceme dlouho informaci utajit, zvolíme délku klíče. Tento typ útoku je velmi často používán a to hlavě díky rostoucímu výpočetnímu výkonu počítačů.
- Luštění se znalostí šifrovaného textu — útočník musí prvně získat několik zpráv zašifrovaných stejným algoritmem a pomocí stejného klíče. Analýzou zašifrovaného textu a jeho aproximací lineární funkcí otevřeného textu může dojít k prolomení šifry. Tato metoda je nazývána lineární kryptoanalýza.
- Luštění se znalostí otevřeného textu — útočník získal nejen zašifrované zprávy, ale i k nim odpovídající otevřený text. Pak už jen stačí najít tajný šifrovací klíč. Vyberou se páry zašifrovaného textu takové, u kterých jejich odpovídající páry otevřeného textu vykazují nějaké rozdíly. Zkoumá se, jak se tyto rozdíly během šifrovacího algoritmu mění. Analyzováním velkého počtu párů lze zjistit správný klíč. Tato metoda je nazývána diferenciální kryptoanalýza.
- Luštění se znalostí vybraných otevřených textů — útočník si muže vybrat otevřený text a jemu odpovídající zašifrovaný text. Pravděpodobnost toho, že získá klíč, je zde mnohem větší.

1.3 Princip kryptografie

Hlavním cílem každého šifrovacího systému je zamaskovat utajovanou zprávu, tak aby byla pro všechny nepovolané osoby zcela nečitelná [27].

Informaci, kterou je nutno zabezpečit označujeme jako otevřený text, proces zabezpečování zprávy pak jako šifrování. Zabezpečený otevřený text se stává šifrovaným textem neboli kryptogramem a sada pravidel použitých pro šifrování otevřeného textu se nazývá šifrovacím algoritmem. Operace tohoto algoritmu se běžně odvíjí od šifrovacího klíče, který společně s textem zprávy představuje vstupní informace pro algoritmus. Chce-li příjemce z kryptogramu obdržet původní zprávu, musí použít dešifrovací algoritmus, který ve spojení s dešifrovacím klíčem převede zašifrovaný text na původní otevřený text [27].

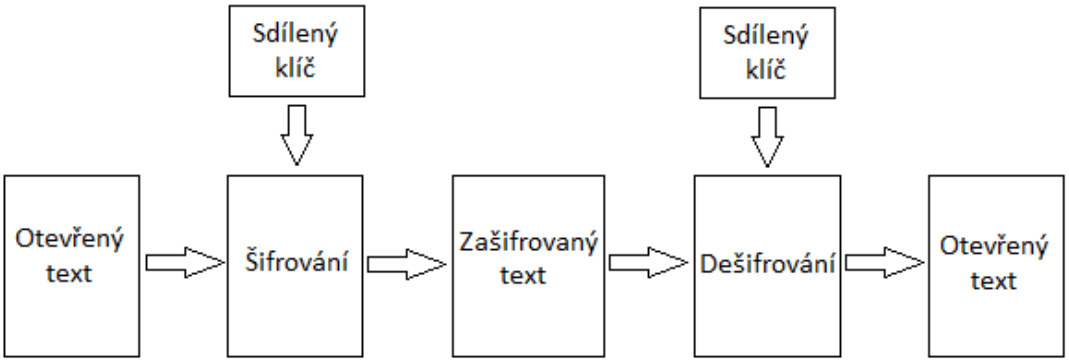


Obr. 1.1: Šifrování a dešifrování [25].

1.4 Symetrické šifrování

Symetrické šifry používají pro šifrování i dešifrování stejný klíč. Odesílatel tedy zašifruje otevřený text a adresát jej dešifruje stejným klíčem jakým byla zpráva zašifrována, proto je potřeba zajistit bezpečné doručení klíče. Pro distribuci klíčů symetrického šifrování je možné využít asymetrického šifrování [18].

Při šifrování se používají takové operace, aby i při znalosti vstupního a zakódovaného textu bylo velmi obtížné odhalit klíč. Bezpečnost závisí především na délce daného klíče. V dnešní době se doporučují používat klíče, které mají délku alespoň 128 bitů [18].



Obr. 1.2: Symetriké šifrování [25].

Symetrické šifry se dělí na dva zakladní typy a to podle toho jak zpracovávají otevřený text:

- Proudové šifry - zpracovávají otevřený text jeden bit po druhém. Využívá se náhodného generátoru, který podle symetrického klíče generuje výstupní hodnoty. Výstup z generátoru je pak kombinován s otevřeným textem [18].
- Blokové šifry - zpracovávají otevřený text po celých blocích dat o stejné délce. Velikost těchto bloků je obvykle 64 nebo 128 bitů a má vliv na bezpečnost algoritmu. Jestliže by velikost bloku byla příliš malá, bylo by možné při daném klíči vytvořit slovník vstupních a výstupních hodnot algoritmu. Tím by byla narušena bezpečnost celého algoritmu [18].

Výhodou symetrické kryptografie je rychlost šífrování i dešifrování a nenáročnost na výpočetní výkon. Nevýhodou pak, že při předávání klíče nezabezpečeným kanálem, může připadný útočník získat tento klíč [11].

Algoritmus DES

Algoritmus DES (Data Encryption Standard) vzniknul v 70. letech a byl později přijat za americkou normu. Tento algoritmus řadíme do kategorie symetrických blokových šifer, protože šifruje blok dat o velikosti 64 bitů. Používá délku klíče 64 b, z toho je 8 bitů kontrolních a 56 b efektivních. Výsledným klíčem je tedy 56-bitové číslo. Postupem času přestal jako standard dostačovat, proto se začal nahrazovat novějšími kryptografickýcmi algoritmy, jako je například 3DES nebo AES [34].

Princip algoritmu je založen na Feistelově struktuře. Výhodou je, že při dešifrování zprávy je postup přesně opačný než při šifrování. Není tedy nutné měnit funkce

ani programovat nové postupy. Stačí, aby vše bylo provedeno v opačném pořadí [11].

Bezpečnost algoritmu: Může nastat situace, že při špatné volbě klíče vypadá kryptogram úplně stejně jako vstupní text. Tento problém je spojen s operacemi uvnitř algoritmu. V dnešní době je algoritmus považován za velice nespolehlivý a nedoporučuje se používat. Je snadno prolomitelný při útoku hrubou silou a to za méně než jeden den. Za to může především délka klíče 56 bitů [11].

Algoritmus AES

Autory algoritmu jsou J. Daemen a V. Rijmen, kteří jej pojmenovali Rijndael. Vznikl jako náhrada za nespolehlivý DES a pomalý 3DES. Je to vítěz soutěže na nový šifrovací standard, který vyhlásil NIST (National Institute of Standards and Technology). Pro jeho název se vžilo označení AES (Advanced Encryption Standard). Platnosti nabyl až roku 2002 a nyní je jedním z nejpoužívanějších symetrických algoritmů [24].

Jde o symetrickou blokovou šifru, kde délku bloku stanovil NIST na 128 bitů, ale reálně lze použít i 192 nebo 256 bitů, s délkou klíče 128, 192 nebo 256 bitů. Samotné šifrování je postupně provedeno ve 4 krocích:

- SubBytes – jednoduchá substituce, kde každý bit je nahrazen jiným podle předem daného klíče. Tím je zaručena nelineárnost šifry a ochrana před jednoduchými algebraickými útoky .
- ShiftRows – v tomto kroku se jednotlivé bity v matici posouvají vlevo a to v každém řádku o jinou hodnotu.
- MixColumns – u této operace dojde k prohození sloupců a poté je vynásoben každý sloupec stejným polynomem.
- AddRoundKey – poslední operace, při které je zkombinován každý byte se subklíčem (ten je získán z původního klíče za pomoci Rijndaelovy tabulky) a po zkombinování bytů subklíče s byty zprávy dostaneme výslednou šifru.

Počet probíhajících kol neboli rund je přímo závislý na délce klíče. Bylo stanoveno 10, 12 a 14 rund pro délky klíčů 128, 192 a 256 bitů. Při modifikaci lze počet rund libovolně změnit [31].

Bezpečnost AES: Z matematického hlediska zatím není znám útok, který by ohrozil bezpečnost algoritmu. V nejbližší době tedy nehrozí jeho prolomení. Předpokládá se že šifra bude bezpečná po dalších 20 až 30 let [11].

Algoritmus IDEA

Algoritmus IDEA (International Data Encryption Algorithm) vzniknul roku 1991 ve švýcarském institutu ETHZ (Swiss Federal Institute of Technology). Jde o blokovou šifru, která používá 128-bitový klíč a šifruje bloky s velikostí 64 bitů. Pro svoji činnost využívá matematické operace modulární sčítaní (modulo 2^{16}) a násobení (modulo $2^{16}+1$) a bitové nonekvivalence (XOR) [24].

Důraz při navrhu byl kladen hlavně na bezpečnost a jednoduchou softwarovou a hardwarovou implementaci, neboť šifrování je rozděleno do 9 rund, z toho 8 je stejných a až poslední runda se liší. Tato vlastnost přispívá k jednoduchosti implementace. Je implementován např. v protokolu SSL nebo PGP [34].

Algoritmus RC4

Je algoritmus vyvinutý kryptologem Ronem Rivestem v roce 1987, ale k odtajnění došlo až v roce 1994. Jedná se o proudovou šifru, kde zpracovávání dat probíhá po bitech nikoli po celých blocích. Velikost klíče je možné zvolit, nejběžněji se volí délky mezi 40 až 256 bity. Jeho hlavní výhoda je rychlost šifrování i dešifrování dat, která je až desetkrát rychlejší než u algoritmu DES. Je tedy vhodný pro softwarovou implementaci. Používá se u aplikací SQL, v protokolu SSL (Secure Sockets Layer) pro zabezpečeni HTTPS (Secure Hypertext Transfer Protocol) komunikace, také u protokolů zabezpečující bezdrátové sítě WEP a novější WPA [31].

Algoritmus se již nedoporučuje používat, protože je snadno prolomitelný a to dokonce na dnes používaných PC během několika minut. Také není odolný vůči útoku „man in the middle“ protože z důvodu rychlosti se neověřuje s kým je komunikováno. Tím dává možnost útočníkovi vydávat se za jednu z komunikujících stran a podvrhnou jeho identitu. I tak je stále používán u protokolů WEP a WPA z důvodu zpětné kompatibility. A také pro šífrování GSM přenosů [11].

Algoritmus HC-128

HC-128 je softwarově orientovaná výkoná synchronní proudová šifra. Je zjednodušenou verzí HC-256 pro 128-bitové zabezpečení. Využívá tedy 128-bitové klíče a inicializační vektor délky 128 bitů. Výhodou je její jednoduchost, bezpečnost, výkon a je volně k dispozici pro jakékoli použití. Algoritmus je vhodný pro moderní mikroprocesory, protože dokáže využít paralelních výpočtů. Díky vysoké paralelnosti dosahuje rychlost šifrování kolem 3 cyklů/byte na procesoru Intel Pentium M. Proto se používa v aplikacách, kde chceme zašifrovat dlouhé proudy dat.

Algoritmus se skládá ze dvou tajných tabulek generovanými z klíče a inicializačního vektoru, každá z tabulek má 512 32-bitových prvků. Pomocí těchto dvou

tabulek je vytvářen keystream. V každém kroku se aktualizuje jeden prvek z tabulky pomocí nelinearní zpětnovazební funkce. Všechny prvky v obou tabulkách jsou aktualizovány každých 1024 kroků.

Fáze algoritmu jsou rozděleny na dvě části. První je inicializace, kde se vytvoří obě pomocné tabulky a provede se 1024 kroků bez vytvoření keystreamu a to z důvodu promíchání tabulek. V druhé části se generuje samotný keystream, v každém kroku je vytvořeno 32-bitové slovo a změněno jedno slovo v tabulce [17].

Algoritmus Rabbit

Rabbit je synchronní proudová šifra, která používá 128-bitový klíč a 64-bitový inicializační vektor. Byla navržena s ohledem na co největší rychlost šifrování, kde na procesoru Intel Pentium dosahuje 3,7 cyklů/byte.

Algoritmus využívá tajný klíč a inicializační vektor v kombinaci s vnitřními stavy bitů a vytváří v každé iteraci 128 bitů dlouhý pseudonáhodný proudový klíč (keystream). Algoritmus se skládá z 513 bitů rozdělených mezi osm 32-bitových stavových proměnných, osm 32-bitových čítačů a jeden přenášející bit. Osm stavových proměnných je aktualizováno ve spojení s nelinearní funkcí. Šifrování se provádí pomocí operace XOR a keystreamu aplikovaného na otevřený text [8].

Algoritmus Salsa20/12

Salsa20 je softwarově orientovaná proudová šifra, která podporuje klíče délky 128 bitů a 256 bitů. Jejím základem je hašovací funkce s 64 vstupních a 64 výstupních bytů. Hašovací funkce se používá v režimu čítače jako proudová šifra. Salsa20 produkuje 64-bytový výstupní blok, který tvoří 32-bytový klíč, 8-bytový nonce (číslo použité pouze jednou), 8-bytový čítač a 16 bytů je určeno dle specifikace.

Salsa20 využívá kombinace tří jednoduchých operací: sčítání modulo 2^{32} , bitové rotace a XOR. Účinné provádění těchto operací v oblasti softwaru nám dává velice dobrý softwarový výkon šifry.

Byly navrženy tři hlavní varianty šifry Salsa20 v závislosti na počtu kol a to Salsa20/8, Salsa20/12 a Salsa20/20. Každý poskytuje určitý kompromis mezi zabezpečením a výkonem. Salsa20/20 se doporučuje pro šifrování v běžných kryptografických aplikacích. Verze Salsa20/12 a Salsa20/8 mají 12 a 8 kol, doporučují se tam, kde se cení vyšší rychlost za cenu menší bezpečnosti. Nejlepší rovnováhu nabízí Salsa20/12, která ma slušný výkon a dostatečně vysokou bezpečnost [7].

Algoritmus SOSEMANUK

SOSEMANUK je synchronní proudová šifra, která má proměnnou délku klíče a to v rozmezí 128 až 256 bitů, inicializační vektor má délku 128 bitů. Bezpečnost šifry je garantována pouze pro 128-bytové zabezpečení. Algoritmus využívá podobné principy návrhů jako proudová šifra SNOW 2.0 a bloková šifra SERPENT. SOSEMANUK se snaží vylepšit nedostatky SNOW 2.0 a to jak po stránce bezpečnosti tak výkonu, např. urychluje inicializační proces.

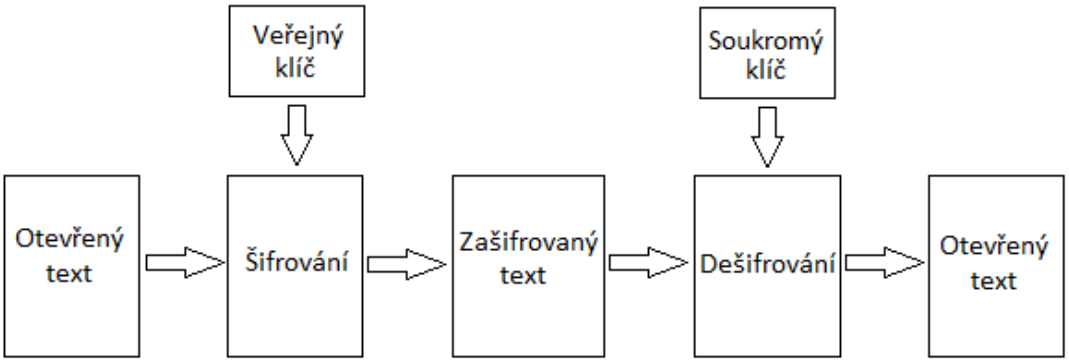
Stejně jako u proudové šifry SNOW 2.0 má SOSEMANUK dvě hlavní části: lineární zpětnovazební registr (LFSR) a konečný stav (FSM). LFSR pracuje s deseti 32-bitovými slovy, v každé kroku je vypočítáno nové 32-bitové slovo. FSM má dva 32-bitové paměťové registry. V každém kroku FSM bere vstupní slovo z LFSR, aktualizuje paměť registru a vytváří 32-bitový výstup. Na každé čtyři po sobě jdoucí výstupní slova z FSM se používá výstupní transformace, založená na blokové šifře SERPENT. Na výsledné čtyři 32-bitová výstupní slova je použita operace XOR se čtyřmi výstupy z LFSR a jsou generovány čtyři 32-bitová slova keystreamu.

Co se týče jeho výkonu v oblasti softwaru, šifruje SOSEMANUK dlouhé datové toky rychlostí 5,6 cyklů/byte na procesoru Intel Pentium M [5].

1.5 Asymetrické šifrování

Asymetrické šifry používají odlišné šifrovací a dešifrovací klíče. Tak jak vidíme na obr. 1.3. Klíče se nesmějí ani rovnat, ani být jeden od druhého odvoditelný. Pro zašifrování zprávy se nejdříve použije veřejný klíč, ten vlastník zveřejní a tím jej může získat kdokoli. Když spolu chtějí dvě strany komunikovat, musí si pomocí dostupných certifikátů získat veřejný klíč protistrany. Pomocí certifikátu se zajistí identifikace a totožnost držitele klíče. Veřejný klíč poslouží k zašifrování komunikace. Po zašifrování komunikace jí není schopen rozšifrovat ani ten, kdo provedl její zašifrování, protože není vlastníkem soukromého klíče toho s kým probíhá komunikace. Tuto komunikaci může rozšifrovat pouze příjemce, jenž je vlastníkem daného soukromeho klíče. Tento klíč by si měl vlastík velice pečlivě hlídat, aby nedošlo k jeho odhalení. V tomto je veliká výhoda proti symetrickým algoritmům, protože odpadá předávání tajného klíče a tím pádem náklady na jejich správu [31].

Výhoda asymetrického šifrování je v tom, že není potřeba posílat tajný klíč a riskovat tak jeho odhalení. Také bývá potřebných daleko méně klíčů, než je tomu u symetrických šifer. Jedné osobě postačuje pouze jeden pár klíčů. Nevýhoda asymetrických šifer je v tom že jsou velice pomalé. Pokud je porovnáme se symetrickými šiframi jsou asi 100 krát pomalejší [18].



Obr. 1.3: Asymetriké šifrování [25].

Velice často se využívá propojení obou druhů šifer a to tak, že jsou z každé šifry využity pouze její výhody. Zpráva se nejdříve šifruje symetrickým klíčem, který se poté zašifruje asymetrickým klíčem. Při dešifrování zprávy se nejdříve pomocí soukromého klíče získá šifrovaný symetrický klíč, kterým se pak dešifruje celá zpráva. Výsledkem tohoto rešení je menší náročnost na výpočetní výkon a také dosažení poměrně veliké bezpečnosti přenosu zpráv [18].

Algoritmus RSA

Algoritmus RSA je jedním z nejrozšířenějších, ale zároveň nejjednoduších algoritmů s veřejným klíčem. Jméno dostal po svých tvůrcích – Ronu Rivestovi, Adi Shamirovi a Leonardu Adlemanovi, kteří jej v roce 1977 vypracovali. Systém může být použit jak pro šifrování, tak i pro digitální podpisy [23].

RSA je považován za dostatečně bezpečný pro dlouhé délky klíčů, dále jsou uvedeny délky klíčů a jejich doporučené použití [34]:

- 512 bitů – nebezpečný. Doporučuje se nepoužívát.
- 768 bitů – průměrně bezpečný. Používat pro méně důležitá data.
- 1024 bitů – bezpečný. Vhodné pro běžné použití.
- 2048 bitů – velice bezpečný. Vhodné pro průmyslové a bankovní nasazení.
- 4096 bitů – vhodný např. pro certifikační autority.

RSA je založen na obtížné faktorizaci velkých čísel. Veřejný a soukromý klíč je

odvozen od dvou prvočísel p a q, která bývají stejně velká kvůli bezpečnosti. Poté je vypočítán jejich součin $n = pq$, dále je zvolen šifrovací klíč e tak, že číslo e musí být nesoudělné s $(p-1)(q-1)$. Nakonec zbývá vypočíst dešifrovací klíč d tak, aby platilo $ed = 1 \bmod (p-1)(q-1)$. Po tomto výpočtu p ani q není potřebné a může dojít k jejich odstranění. Čísla e a n slouží pro veřejný klíč a číslo d pak jako tajný klíč [34].

Pro šifrování použijeme vztah:

$$c_i = m_i^e \bmod n.$$

a pro dešifrování platí vztah:

$$m_i = c_i^d \bmod n.$$

Pro jednotlivé číselné bloky m_i musí platit $m_i < n$. Zprávu lze také zašifrovat číslem d a dešifrovat číslem e [34].

Algoritmus RSA je oproti symetrickému algoritmu DES při hardwarové implementaci až 1000 krát pomalejší a při softwarové implementaci až 100 krát pomalejší [31].

Bezpečnost algoritmu závisí především na nemožnosti faktorizace čísla n, které je součinem p a q. Pokud tyto prvočísla nesprávně zvolíme, má to za následek, že systém není bezpečný a útočník jej může prolomit. Dnes jako vhodnou délku n volíme 2048 bitů tedy číslo 2^{2048} a velikost tajného klíče stejné délky [36]. Šifra je také zranitelná některými typy útoků (např. útok s vybraným textem) a je tedy zavislá na správné implementaci [34].

Algoritmus Diffie–Hellman

Tento algoritmus vynalezli v roce 1976 pánové Whitfield Diffie a Martin Hellman. Je to rozšířený asymetrický algoritmus pro výměnu klíčů. Nepoužívá se tedy pro šifrování dat jako takových, ale jako protokol výměny klíčů pro symetrické šifry [11]. Jeho bezpečnost je založena na metodě řešení výpočtu diskrétních logaritmů, které jsou výpočetně podobné faktorizaci velkých čísel. Algoritmus se využívá v nechráněném prostředí, kde si obě strany vygenerují vlastní klíče, ty si vymění a na jejich základech si odvodí klíč tajný pro vlastní přenos. Tento postup je složitější a ubírá na funkčnosti vlastního algoritmu, neboť se jim nedá podepisovat [31, 34].

Předností algoritmu je, že případný útočník, který odposlouchává komunikaci není schopen zachytit tajný klíč, protože tento klíč není nikdy posílán v otevřené formě. Velkou nevýhodou je prolomení útokem „Man in the middle“, protože nedochází k ověřování komunikujících stran. Tento protokol je tedy vhodný tam, kde případný útočník nemůže vstoupit mezi komunikující strany [11].

Algoritmus El Gamal

Tento algoritmus vzniknul v roce 1984 a dostal jméno po svém tvůrci Taheru Elgamalovi. Je velice podobný již výše popsanému algoritmu Diffie–Hellman, jeho bezpečnost je také závislá na metodě řešení výpočtu diskrétních logaritmů. Ale na rozdíl od něj je vhodný i pro šifrování a dešifrování zpráv. El Gamal bývá často používán pro digitalní podepisování [11]. Využívá se především pro systémy GNU (General Public License), PGP (Pretty Good Privacy), GPG (GNU Privacy Guard) a další [31].

Hlavní nevýhodou je, že zašifrovaná zpráva má dvojnásobnou délku než zpráva nezašifrovaná. To je nejspíše i důvod proč není tak rozšířen a používán jako algoritmus RSA [36].

Bezpečnost algoritmu byla ve své době na velice dobré úrovni, ale i dnes má některé vlastnosti, které jsou výhodné. Ovšem jako celek není považován za úplně bezpečný, zejména pro podepisování není vhodný. K šifrování zpráv lze bezpečně používat, ovšem jak již bylo zmíněno výše, přenášená zpráva má dvojnásobnou délku, proto se v praxi příliš nepoužívá. Výhoda oproti jiným algoritmům je v použití náhodného parametru x pro výpočet bloku zpráv při šifrování. Tento parametr je pro každý blok jiný a tato vlastnost je zásadní, protože algoritmy typu RSA tuto vlastnost nemají a obsahují tak bezpečnostní slabinu, která se odstraňuje dodatečne. Naopak, kdyby se parametr v implementaci neměnil došlo by k oslabení algoritmu a jeho snadnému prolomení [21].

Algoritmy založené na eliptických křivkách

Algoritmy založené na eliptických křivkách (ECC) spadájí do kategorie asymetrického šifrování, kde šifrování a dešifrování probíha s využitím eliptických křivek. Nevýhoda asymetrických algoritmů je ta, že bezpečná velikost klíče je daleko větší než u symetrických algoritmů a tím pádem jsou značně pomalejší. V tab. 1.1 jsou uvedeny velikosti klíčů vybraných algoritmů v závislosti na srovnatelné bezpečnosti.

Tab. 1.1: Srovnání velikosti klíčů vybraných algoritmů [13].

Symetrické šifry	ECC	DH/RSA
56	105	417
80	163	1024
128	283	3072
192	409	7680
256	571	15360

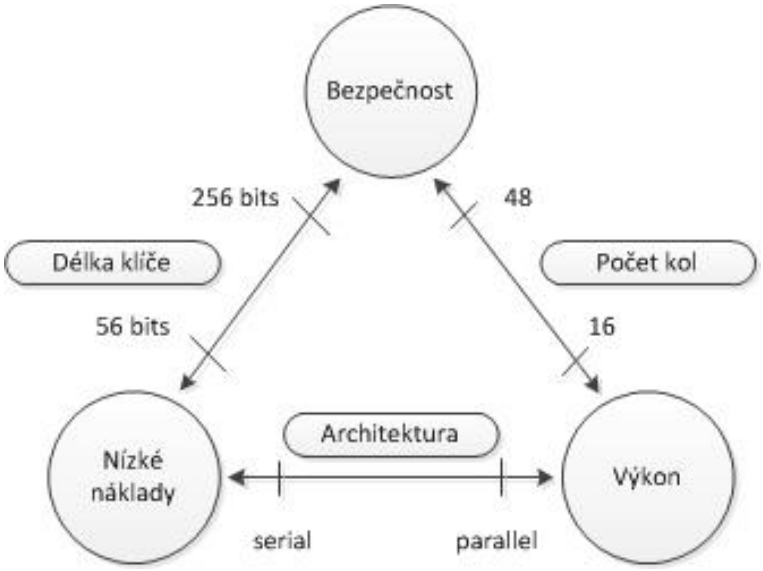
V roce 1985 jako první navrhli použití eliptických křivek pro návrh asymetrických algoritmů pánové Victor Miller a Neal Koblitz. Jde vlastně o analogii k již existujícím systémům, kde je nahrazena modulární aritmetika arimetikou operací s body na eliptické křivce. Z důvodů kratších klíčů lze systémy založené na eliptických křivkách používat na slabším hardwaru. Hodí se tak především pro implementaci na čipové karty a další podobná zařízení. V současné době se staly eliptické křivky alternativou k již zavedeným algoritmům. Výhoda je hlavně v rychlosti a malé náročnosti na hardware a software, ale nasazování je však pomalejší a to nejspíše z důvodu, že klasické asymetricke algoritmy jsou známy a používány delší dobu [26].

Bezpečnost algoritmu spočívá v obtížném řešení úloh diskrétního logaritmu pro eliptické křivky. V dnešní době je tato úloha daleko složitěji řešitelná než výpočet normálního diskrétního logaritmu. Eliptické křivky jsou zranitelné při použití kvantových algoritmů, například Shorova algoritmu [26, 36].

2 LEHKÁ KRYPTOGRAFIE

Lehká kryptografie je poměrně mladé vědní odvětví, které se zaměřuje na nové konstrukce, adaptaci nebo efektivní implementaci kryptografických primitiv a protokolů pro výpočetně omezená zařízení jako jsou např. RFID, low-end čipové karty, mikrokontroléry atd. Tato zařízení neumožňují implementovat klasickou kryptografii, jako například AES s délkou klíče 256 bitů, či klasická asymetrická kryptoschémata např. RSA apod. Musíme totiž počítat s omezeným množstvím energie a velikostí samotných čipů [30].

Při návrhu se proto musíme vyrovnat s kompromisem mezi bezpečností, náklady a výkonem, tak jak je zobrazeno na obr. 2.1.



Obr. 2.1: Schéma kompromisů pro lehkou kryptografii [30].

Obvykle každé dva ze tří cílů návrhu, tedy bezpečnost a nízké náklady, bezpečnost a výkon, nebo nízké náklady a výkon lze snadno optimalizovat, ovšem je velmi obtížné optimalizovat všechny tři cíle návrhu současně. Například bezpečného a vysoce výkonného hardwaru je dosaženo zřetězením architektury, která současně obsahuje i řadu protiopatření proti útokům postranními kanály, výsledná konstrukce však bude mít vysoký plošný požadavek, tedy vysoké náklady. Na druhé straně je možné navrhnout bezpečný algoritmus s nízkými náklady, ale s omezeným výkonem [30].

Informace chráněné pomocí čipů RFID, jsou často omezeny buď svojí cenou, časem utajení, nebo časem jaký utočník potřebuje na provedení útoku. Právě toho využívá lehká kryptografie, která stanovila požadovanou bezpečnost na 80 bitů. Toto číslo se může zdát příliš nízké, ale vzhledem k chráněné informaci je to více než

dostačující. Případný útočník by musel vynaložit příliš velké prostředky, nebo úsilí, aby 80 bitový klíč prolomil. A to i v dnešní době výkonných výpočetních zařízení, přičemž zisk z tohoto útoku by nebyl zcela adekvátní.

Pro potřeby lehké kryptografie byly vyvinuty blokové a proudové šifry a také hašovací funkce pro čipy RFID s 80-bitovou bezpečností [19].

2.1 Blokové šifry

Během šifrování dojde prvně k rozdělení zprávy na bloky s pevným počtem bitů. Výhodou je jejich bezpečnost, která je vyšší než u proudových, ale oproti nim jsou daleko pomalejší.

U blokových šifer využiváme dva hlavní principy. Prvním je Feistova struktura jejímž představitelem je např. DES, který je popsán zde 1.4 a druhým substitučně permutační struktura, jejíž představitelem je AES popsaný zde 1.4 Pro čipy RFID byly vyvinuty nové algoritmy, např. z principu AES evropská šifra PRESENT a z principu DESu čínska šifra LBLOCK. AES, DES, PRESENT a LBLOCK mají některé společné prvky, pracují v cyklech, kde výstup z jednoho cyklu je vstupem do dalšího. V každém cyklu se pak na vstup přičítá tzv. rundovní klíč a výsledek vstupuje do substitučních boxů (S-boxů), přitom se za S-boxy zařazuje permutace bitů, tak aby v dalším cyklu každý bit výstupu vstoupil do jinného S-boxu. Všechny tyto algoritmy také používají úpravu šifrovacího klíče, proto aby do každého cyklu vstupoval jiný rundovní klíč. Z tohoto vyplívá, že každé principielní schéma těchto blokových šifer má dvě částí, první pro přípravu rundovních klíčů a druhou pro zpracování dat [35].

S-boxy v RFID čipech jsou jediným nelineárním prvkem u zmíněných blokových šifer. Na PC je realizace provedena pomocí konstantních tabulek, ale u RFID je však málo paměti a velké šifry jako EAS či DES se do těchto malých zařízeních nevejdou. U EAS jsou S-boxy typu 8x8 u DES 6x4, proto se u šifry PRESENT i LBLOCK používají S-boxy typu 4x4. Musíme však stále dbát na nelinearitu, protože při zajištění příliš malé nelinearity je šifra méně bezpečná. Složitost pak lze ještě dohnat počtem cyklů, kde PRESENT i LBLOCK má 32 rund, AES 10 (12, 14) rund a DES 16 rund [35].

V tab. 2.1 jsou srovnány obě šifry a některé další z pohledu potřebného místa (jedním GE - rozumíme jeden prvek neboli hradlo) a výkonu.

Tab. 2.1: Charakteristiky některých blokových šifer pro RFID [35].

Bloková šifra	Délka bloku [bit]	délka klíče [bit]	Plocha [počet GE]	Rychlost [kbit/s]@100 kHz
PRESENT	64	80	1570	200
LBlock	64	80	1320	200
KATAN	64	80	1054	25
DES	64	56	2300	44
DESXL	64	184	2168	44
mCrypton	64	128	2500	492
HIGHT	64	128	3048	188
XTEA	64	128	3490	57

2.2 Hashovací funkce

Hashovací funkce je základní nástroj k zajištění bezpečnosti, protože zajišťuje autentičnost a integritu digitálních dokumentů, souborů a dat přenášených nejrůznějšími komunikačnímy protokoly. Můžeme říci, že je to jednosměrná matematická funkce, která na vstup přijímá řetězec znaků libovolné délky a výsledkem je pak řetězec znaků s pevnou délkou, tzv. digitální otisk. To nám zaručuje neporušenost dat. Dnes se za bezpečné považují otisky délky 256 bitů, nebo 512 bitů, ale postačí i 128-bitové [20].

Zeslabená hash takzvaná lehká hash má poskytnou maximální míru bezpečnosti, která je realizovatelná v čipech RFID. Jako nejlepší se jeví návrh jménem PHOTON. PHOTON-128 zabírá pouze velmi malou plochu (1122 GE) a přitom zajišťuje velkou bezpečnost (otisk 128 bitů). Přitom například pro SHA-1 je potřeba 5527 GE, pro SHA-2 10868 GE a pro SHA-3 nad 12000 GE, což je pro RFID čipy příliš [20].

PHOTON nemá velkou konkurenci, protože ostatní návrhy hashovacích funkcí pro RFID zabírají větší plochu a mají menší bezpečnost, přesné charakteristiky jsou v tab. 2.2.

Tab. 2.2: Charakteristiky některých hashovacích funkcí pro RFID [20].

Hash	Délka hashe [bit]	Bezpečnost [vzor]	Bezpečnost [kolize]	Plocha [počet GE]	Rychlost [kbit/s]
PHOTON-80/20/16	80	2^{64}	2^{40}	865	1,51
DM-PRESENT	64	2^{64}	2^{32}	1600	5,85
PHOTON-128/16/16	128	2^{112}	2^{64}	1122	0,69
PHOTON-160/36/36	160	2^{124}	2^{80}	1369	1,03
PHOTON-256/32/32	256	2^{224}	2^{128}	2177	0,88

2.3 Proudové šifry

Zpracovávání dat probíhá po jednotlivých bitech a je velice rychlé. Mezi hlavní výhody patří, že nedochází k šíření chyb nebo jen velmi málo. Proudové šifry můžeme rozdělit na dvě základní skupiny, buď synchronní nebo asynchronní a to podle toho jakým způsobem generují klíčový proud znaků tzv. keystream [12].

- Synchronní – jsou to takové šifry, kde je keystream generován nezávisle na nešifrovaném nebo šifrovaném textu. Generování keystreamu je závislé jen na klíči a šifrovacím algoritmu. Z toho plyne, že odesílatel i příjemce zprávy musejí být synchronizováni, tedy musí sdílet stejný klíč i aktuální stav algoritmu. Pukud dojde ke ztrátě synchronizace, nepůjde bez speciálních metod zprávu dešifrovat. Pokud dojde během přenosu ke změně nějakého znaku v textu, nebude ovlivněn zbytek zprávy. Tohle je velká výhoda v situacích, kdy během přenosu dochází k velkému počtu chyb.

- Asynchronní – jsou takové, kde je keystream generován pomocí klíče a několika předchozích znaků šifrovaného textu. I zde by měli být odesílatel a příjemce synchronizováni, ovšem pokud dojde k výpadku, měla by se šifra po několika znacích znovu sama synchronizovat. Pokud dojde během přenosu ke změnění nějakého znaku v textu, může dojít ke změně dešifrování pouze několika následujících znaků a zbytek zprávy by neměl být ovlivněn.

Jak už bylo řečeno proudové šifry jsou daleko rychlejší a také méně hardwarově náročné než blokové. Ovšem pokud nejsou při návrhu dodržena určitá bezpečnostní opatření mohou se stát lehkým cílem kryptografických útoku, např. by se neměl použít dvakrát stejný výchozí stav, klíče by se neměly opakovat, šifrovací klíč by měl mít velikost periody a musí být nemožné jeho obnovení nebo zjištění vnitřního stavu šifrovacího klíče atd [12].

Algoritmus Grain v1

Grain v1 je hardwarová synchronní proudová šifra. Používá 80-bitový tajný klíč a 64-bitový inicializační vektor. Jeho novější verze byla posílena s ohledem na bezpečnost a využívá 128-bitový tajný klíč a 80-bitový inicializační vektor. Návrh šifry co nejvíce dbá na omezené prostředky použitého hardware jako např. počet hradel, spotřeby elektrické energie či velikost paměti.

Konstrukce je založena na dvou posuvných registrech, jeden s lineární zpětnou vazbou (LFSR) a druhý s nelineární zpětnou vazbou (NFSR). Oba posuvné registry mají 80-bitovou velikost. LFSR zaručuje minimální periodu pro keystream a tím poskytuje bezpečnost výstupu. Ve funkci filtru zde vystupuje NSFR, který nelinearizuje a zároveň kumuluje zpětnou vazbu z lineárního registru přes nelineární funkci. Po inicializaci začíná šifra generovat heslo jako bitový proud, který je načítán operací XOR na proud bitů nešifrovaného textu [16].

Algoritmus MICKEY 2.O

MICKEY 2.0 (Mutual Irregular Clocking KEYstream generator) je synchronní proudová šifra určená pro hardwarové platformy s omezenými zdroji. Má zajistit minimální hardwarovou složitost a poskytnout vysokou úroveň bezpečnosti. Přitom používá 80-bitový klíč a inicializační vektor s délkou až 80 bitů.

Šifra se skládá ze dvou 100-bitových posuvných registrů, jeden je lineární a druhý nelineární, každý z nich je nepravidelně taktován. Specifické časové mechanismy přispívají ke kryptografické síle, přičemž zajišťují požadavky na periodu a pseudonáhodnost, tak aby se minimalizovala úspěšnost kryptografických útoků.

Specifikace šifry udávají, že lze generovat až 2^{40}-bitový keystream z každého tajného klíče a inicializačního vektoru. Byla také navržena rozšířená varianta zvaná MICKEY-128 2.0, která má 128-bitový klíč a IV až do velikosti 128 bitů.

MICKEY 2.0 může být realizován na mimořádně malých rozměrech hardwaru, což je vhodné tam, kde jsou hlavní požadavky na nízký počet hradel nebo nízkou spotřebu. Nepravidelné hodinové pulzy způsobují, že šifru nelze snadno paralelizovat a tak je její rychlost šifrování v softwarové oblasti nižší [2].

Algoritmus Trivium

Trivium je hardwarová synchronní proudová šifra, která poskytuje kompromis mezi výkonem a složitostí. Při návrhu tohoto algoritmu bylo zkoumáno jak až může být šifra zjednodušena, aniž by byla ohrožena její bezpečnost, rychlost a flexibilita.

Šifra dokáže generovat až 2^{64}-bitový keystream z 80-bitového tajného klíče a 80-bitového inicializačního vektoru (IV). Vnitřní stav algoritmu obsahuje 288 bitů, skládající se ze tří vzájemně propojených zpětnovazebních registrů o délkách 93, 84 a 111 bitů. Jako většina proudových šifer se šifrování skládá ze dvou fází. V první je inicializován počáteční stav pomocí klíče a inicializačního vektoru (IV) a jsou provedeny 4 cykly s 1152 iteracemi. V druhé fázi je předchozí stav opakovaně aktualizován a slouží ke generování keystreamu. Pomocí operace XOR aplikujeme keystream na otevřený text.

Proudová šifra Trivium byla navržena jako kompaktní v omezených prostředích a rychlá v aplikacích, které vyžadují vysoký výkon. Zejména základní rychlost šifrování může být zlepšena pomocí paralelizace (umožňuje výpočet 64 iterací najednou), bez zbytečného zvýšení plochy čipu nezbytné pro jeho provedení. Ačkoli Trivium necílí na softwarové použití, šifra je poměrně účinná i na standardních PC [9].

Algoritmus Edon80

Edon80 je hardwarová synchronní proudová šifra využívající 80 bitů dlouhého klíče a 64 bitů dlouhého inicializačního vektoru.

Vnitřní strukturu lze chápat jako zřetězenou architekturu 80 jednoduchých 2-bitových transformátorů tzv. e-transformátorů. Edon80 pracuje ve třech různých režimech.

V prvním režimu tzv. keysetup módu se nastavuje klíč. A to tak, že se 80-bitový klíč prakticky rozdělí na čtyřicet 2-bitových částí. Podle nich se potom určí, které ze čtyř předem daných operací proběhnou v každém e-transformátoru.

V druhém režimu tzv. IV setup módu se pomocí klíče a inicializačního vektoru nastaví e-transformátory. Na začátku jsou všechny e-transformátory vypnuty, poté probíhá 80 cyklů, kde jsou postupně všechny e-transformátory zapínány a je měněna jejich 2-bitová proměnná.

V posledním režimu tzv. keystream módu je vytvářen samotný keystream. Po uplynutí 80 cyklů začne z posledního, tedy osmdesátého e-transformátoru proudit požadovaný keystream. Pro bezpečnost je důležité, že je použit každý druhý vystupní bit. Nakonec je keystream aplikován na otevřený text pomocí operace XOR [14].

Algoritmus Decim v1

Decim je opět synchronní proudová šifra určená pro hardware s omezenými zdroji. Používá 80-bitový tajný klíč a 64-bitový inicializační vektor.

Nejdůležitější částí je zpětnovazební posuvný registr (LSFR) délky 192 bitů. Ten je definován primitivním zpětnovazebním polynomem. Další částí je jedna logická funkce o sedmi proměnných. Jejím vstupem je 7 bitů z LSFR. Funkce je použita vždy dvakrát v každém kroku, tedy vygeneruje dvojici bitů a ty jsou poté skládány do bitového proudu. Poslední částí je dělící mechanismus ABSG, který bitový proud transformuje na kratší z důvodu ochrany výstupního keystreamu.

Algoritmus má dva stavy. První je inicializace, kde se z tajného klíče a inicializačního vektoru určí nové hodnoty LSFR. K tomu je potřeba alespoň 192 cyklů, aby se LSFR choval zcela náhodně. V dalším stavu dochází ke generování samotného keystreamu a to tak, že z bitů LSFR se pomocí logické funkce vypočítají 2 bity a odešlou se do ABSG. Poté se pozmění hodnoty v registru LSFR a vypočítají další dva bity. Z příchozích bitů vytváří ABSG konečný keystream [4].

Algoritmus F-FCSR-H

F-FCSR-H je hardwarově orientovaná proudová šifra. Využívá 80-bitový tajný klíč a 32 až 80 bitů dlouhý inicializační vektor.

Hlavní část algoritmu tvoří posuvný registr se zpětnovazebním přenosem (FCSR) s délkou 160 bitů. Je to alternativa k LFSR, zatímco LFSR používá jednoduché bitové sčítání, FCSR navíc přidává bity carry. Díky tomuto je funkce nelinearní, přesněji kvadratická. Proto můžeme keystream generovat přímo z vnitřního stavu algoritmu.

Dnes již tento algoritmus není považován za bezpečný, protože na běžném PC jej lze pomocí kryptoanalitického útoku prolomit během několika sekund [6].

3 IMPLEMENTACE ALGORITMŮ

Jak se dá předpokládat bývá hardwarová implementace daleko rychlejší než softwarová. U hardwarové implementace můžeme totiž zpracování dat provádět paralelně (provádět více procesů zároveň). Pokud dojte k odhalení nějaké mezery či chyby v návrhu algoritmu je však úprava hardwaru velice náročná a drahá, tedy systém se stává neflexibilní. To neplatí u softwarové implementace, kde můžeme snadněji modifikovat danou implementaci a snížit tak náklady [3]. V následujících podkapitolách budou podrobněji popsány výhody a nevýhody obou implementací.

3.1 Útok postranními kanály

U všech kryptosystému je jednou z nejdůležitějších věcí implementace vybraného algoritmu či protokolu. Ve stutečnosti je pouze malá část provedených útoků zaměřena na princip algoritmu nebo protokolu. Větší část je uskutečnena na implementaci. Pokud je algoritmus softwarově nebo hardwarově špatně implementován, lze na něj útočit různými druhy útoků [28]. Mezi ně patří např. útok postranními kanály, který je posán dále.

Útoky postranními kanály se nesoustředí na hledání slabého místa ve struktuře algoritmu, ale využívají informací, které jsou dány z fyzické implementace při konání operací daného algoritmu. Tato metoda byla objevena teprve nedávno a znamená jisté riziko pro bezpečnost kryptografie. Můžeme mluvit o různých typech postranních kanálů, každý je založen na nějaké měřitelné informaci. Za hlavní typy postranních kanálů jsou považovány elektromagnetický, časový, chybový a proudový kanál, ale existují i další. Tyto postranní kanály mají vysokou účinnost při útoku na implementaci [29].

Jako příklad lze uvést tzv. „časový útok“ na algoritmus RSA. Tento algoritmus je blíže popsán v kapitole 1.5. Jde o to, že klíčem je velké číslo a při dešifrování dochází k umocnění zprávy na toto velké číslo. Tato operace může trvat dlouhou nebo kratkou dobu a to v závislosti na tom, jaké hodnoty bitů nabývá soukromý klíč. Taková „drobnost“ stačí k omezení prostoru klíčů a tedy k prolomení algoritmu. Časový útok lze podobným způsobem použít u dalších asymetrických algoritmů např. DSA, Diffie–Hellman [29].

3.2 Hardwarová a softwarová implementace

Hardwarová implementace

Donedávna se vyskytovaly veškeré šifrovací produkty ve formě specializovaného hardwaru zapojeného do komunikačního kanálu, tímto kanálem proudila všechna šifrovaná data. Přestože je dnes softwarové šifrování velmi rozšířené, hardware je stále používán pro vojenské a důležité komerční aplikace [32]. Hlavní důvody pro použití hardwarového šífrování jsou následující:

- První z nich je určitě rychlost. Jak víme z předchozích kapitol, algoritmy se skládají z mnoha různých složitých operací, které přemění vstupní zprávu na zašifrovaný text. Ve většině případů takové operace nejsou běžně zabudované v procesorech počítačů, proto na nich algoritmy běží pomale a neefektivně. Někteří kryptografové se snažili, aby jejich algoritmy byly vhodnější pro softwarovou implementaci, ale specializovaný hardware bude vždy v šifrování dat rychlejší [32].
- Druhým důvodem je zajisté bezpečnost. Šifrovací algoritmus, který běží na běžném počítači nemá téměř žádnou fyzickou ochranu. Proto může útočník použít různé nástroje na pozměmění algoritmu, aniž by to někdo zjistil. Naopak zařízení pro hardwarové šifrování lze bezpečně zapouzdřit do boxů a ochránit tak algoritmus od možné modifikace. Každé elektronické zařízení vyzařuje do prostoru elektromagnetické záření, které může odhalit co se děje uvnitř zařízení. Z toho důvodu jsou boxy odstíněny, aby nemohlo dojít k úniku citlivých informací. Běžné počítače mohou být také odstíněny, ale neděje se tak z důvodu složitosti [32].
- Poslední důvod proč zvolit hardwarovou implementaci je snadnost její instalace. Šifrují se hlavně telefoní hovory, faxové přenosy, emaily, citlivá data atd. Takové šifrování je pro běžné uživatele neviditelné a levnější než nasazení softwarové implementace [32].

Dnes je k dostaní několik druhů hardwarových šifrovacích zařízení. Například šifrovací moduly pro ověřování hesla nebo správ klíčů banky. Dále šifrovací boxy pro komunikační spojení, nebo také desky, které se dají připojit do osobních počítačů. Ovšem tato zařízení mají i své nevýhody. Pokud si chceme pořídit takovéto zařízení pro konkrétní účel, musíme pečlivě zkontrolovat omezení jako např. podpora hardwaru počítače, operačního systému, aplikačního software a tak dále. Desky určené do běžných počítačů obvykle šifrují všechna data zapisovaná na harddisk, tyto desky nebývají odstíněny proti elektromagnetickému záření, což není zcela bezpečné [32].

Útoky na hardwarovou implementaci

Útoky na zařízení můžeme přesně rozdělit podle toho, zda máme zařízení fyzicky v držení, nebo jedná-li se o útok spíše softwarový (vzdálený). Softwarový útok je podobný klasickým útokům, jde o objevení softwarové chyby, při které získáme data i bez znalosti hesla nebo pinu. Pokud máme zařízení k dispozici můžeme využít několik druhů útoku. Ty se liší obtížností a náročností na vybavení útočníka [22]:

- Neinvazivní metody – jsou nejméně náročné, spočívají zejména ve změně provozních podmínek, aby se zařízení chovalo jiným způsebem, než jakým je běžné. Existuje mnoho druhů neinvazivních útoků. Např. změnou teploty, buď podchlazení nebo přehřátí.
- Semiinvazivní metody – jsou obtížnější, ale velice účinné. Zářízení je rozebráno jen částečně a je na něj působeno nějakým druhem záření, nejčastěji elektromagnetickým nebo silným světelným zdrojem.
- Invazivní metody – jsou nejsložitější a nejnáročnější na vybavení. Útočník zařízení rozebere až na samotný čip a pomocí speciálního hardwaru, mikroskopů a mikrosond se napojí na sběrnici, nebo čte data z paměti.

Softwarová implementace

Všechny šifrovací algoritmy mohou být implementovány softwarově. Hlavní nevýhody jsou v rychlosti, snadné změně nebo manipulaci pro případného útočníka. Výhodou je flexibilita, přenositelnost, snadnost použití a také jednoduchá aktualizace. Algoritmy psané např. v jazyku C mohou být realizovány s malou modifikací na libovolném počítači. Mohou být levně kopírovatelné a začleněny do větších aplikací jako jsou komunikační programy a jiné [32].

Programy pro šifrování jsou velmi populární a dostupné pro všechny hlavní operační systémy. Používají se k ochraně souborů, kde uživatel může data libovolně šifrovat i dešifrovat. Ovšem je důležité, aby byla spravá klíčů dostatečně zabezpečena. Klíče by neměly být uloženy na disku nebo v paměti počítače bez ochrany. Po skončení šifrování by měly být klíče a původní nezašifrované soubory smazány. Na to ale mnoho programů nedbá a může tak byt ohrožena bezpečnost zašifrovaných dat. Samozřemě případný utočník může proniknout k samotnému počítači a měnit nebo modifikovat šifrovací program a získat tak data [32].

3.3 Hardwarová zařízení s omezeným výpočetním výkonem

Jsou to zařízení, která disponují malým výpočetním výkonem a malou kapacitou paměti.

3.3.1 Smart karty

První skupinou jsou tzv. smart karty neboli chytré čipové karty. Ty obsahují integrovaný obvod, který obsahuje mikroprocesor, vstupní a výstupní rozhraní, paměti RAM, ROM a EEPROM. V některých kartách muže být implementován i kryptografický procesor, který provede kryptografické výpočty rychleji než standardní mikroprocesor. Smart karty mají širokou škálu použití a můžeme je rozdělit do několik velkých skupin [1]:

- Jednoduché smart karty (simple file-system-oriented smart cards without public key capability) – Jsou orientované na systém souborů bez veřejného klíče. Podporují jen symetrické algoritmy např. DES nebo 3DES.
- Pokročilé smart karty (advanced file-system smart cards with public key capability) – Jsou zaměřeny na systém souborů s veřejným klíčem. Obsahují v sobě privátní klíče a přiřazené certifikáty.
- Java karty – dovolují vytvoření uživatelských příkazů na kartě. Tyto příkazy jsou realizovány pomocí upraveného programovacího jazyka Java.
- Windows-powered smart karty – Jsou karty s operačním systémem „Windows for Smart Card“, které dovolují realizaci uživatelských příkazů. Tento typ karet vyvinula společnost Microsoft.
- MULTOS smart karty (multi-application operating system) – Tyto karty poskytují rozhraní systému souborů a také vypracované prostředí pro uživatelské aplikace.

Když se na smart karty zaměříme z pohledu softwarového tak obsahují: Operační systém, aplikace a zavaděč. Operační systém se stará o řízení komunikace mezi aplikacemi a čipem. Aplikace jsou naprogramovány v jazyce podle operačního systému. Zavaděč slouží k nahrávání a odstraňování aplikací [15].

3.3.2 RFID čipy

Tyto čipy mohou být různě velké, zde máme na mysli hlavně ty nejmenší, které mají maximálně 10000 GE (GE - rozumíme jeden prvek neboli hradlo). Velmi tvrdé

požadavky na lehkou kryptografii v čipech RFID zahájily výzkum nových standardů. V čipech RFID je na krytografii vymezeno pouze 1000 až 2000 hradel, ve vyjímečných situacích i více, z celkového počtu pro celý čip. Pro využití blokových šifer v RFID byly vyvinuty např. algoritmy PRESENT a LBLOCK, jejichž výkon a plocha jsou uvedeny v kapitole 2.1. Pro využití hash funkce v RFID byl vyvinut např. PHOTON jehož charakteristiky jsou zde 2.2. Pro proudové šifry byly vyvinuty v RFID např. Trivium, Grain v1 a MICKEY 2.0 a další, jejíchž softwarový výkon je otestován zde 4.2. V budoucnu snad budou vyvinuta i asymetrická schémata pro RFID čipy, což je ale značně náročné [35].

Omezení daná rozměry a napájením těchto čipů, dala vzniknout jak pasivním tak aktivním čipům. Pasivní pro svoji práci využívají energii z vysílače (nemají vlastní baterie) a aktivní čipy vlastnící baterie mohou osahovat i složitější procesory.

Fyzikální limity čipů RFID jsou:

- Rozměr zařízení – je to rozměr zařízení, který limituje množství přijaté a vysílané energie (kondenzátory, antény).
- Rozměr čipu a použitá technologie – omezuje množství hradel (je nutné, aby se zde podařilo vtěsnat nejenom „procesor“, ale i paměť a další komponenty).
- Velikost použitelné energie – u pasivních RFID čipů omezuje využitelnost pro výpočty.

3.3.3 Mikrokontroléry

Díky jejich univerzálnosti, malé spotřebě a nízké ceně jsou využívány v mnoha elektronických zařízeních. Mikrokontroléry s kryptografickými prostředky využíváme např. v čidlech, senzorech, bezdrátových senzorových sítích, u průmyslových kontrolních a řídících prvků, dále v nejrůznějších systémech ochran proti krádeži atd.

Můžeme je označit jako jednočipové mikropočítače. Obsahují v jediném pouzdře všechny podstatné části mikropočítače: Řadič a aritmetickou jednotku, paměť programu, ta je buď typu EPROM, flash, nebo ROM, paměť dat typu R/W, někdy doplněnou EEPROM, periferní obvody pro vstup a výstup dat.

Dále obvykle mikrokontroléry obsahují generátor hodinového signálu a další technické prostředky, jako jsou obvody pro kontrolu správné činnosti mikrokontroléru, obvody pro programování kódové paměti přímo v aplikaci, A/D a D/A převodníky, řadiče přerušení, DMA řadiče apod.

Mikrokontrolér může být rozšířen o periferní obvody, jako jsou: paraelní IO porty, seriové rozhraní, čítače a časovače.

Všechny typy mikrokontrolérů mají svoji instrukční sadu. Ta obsahuje seznam strojových instrukcí, které jsou psány v assembleru. Z důvodu, že je v každém typu mikrokontroléru použita jiná instrukční sada assembleru, není možné přenášet tyto programy mezi různými mikrokontroléry. Z tohoto důvodu se pro vytvoření zdrojových kódů používá univerzální programovací jazyk C. Tento jazyk nižší úrovně je lépe přenositelný mezi různé architektury.

Podle typu mikrokontroléru do něj můžeme implementovat různé šifrovací algoritmy např. AES, DES, 3DES, RSA, ECC a dále algoritmy pro výpočet hash např. MD5 či SHA-1 [15].

4 TESTOVÁNÍ PROUDOVÝCH ŠIFER

4.1 Způsob testování

Otestovány byly proudové šifry uvedené v kapitole 1.4. Přesněji tedy byly vybrány šifry HC-128, Rabbit, Salsa20/12, SOSEMANUK, ty jsou vhodné pro softwarové aplikace s vysokými požadavky na propustnost. A všechny šifry z kapitoly 2.3 vhodné pro hardwarové aplikace s omezenými zdroji tedy Grain v1, MICKEY 2.0, Trivium, Edon80, Decim v1 a F-FCSR-H.

Testování probíhalo na proceseru Intel Core 2 Duo, přesné označení je T5800, které pracovalo na frekvenci 2 GHz. Operační systém byl vybrán Linux Ubuntu 12.04 a použitý kompilátor gcc-4.6. Pro účely testování byl použit volně přístupný testovací nástroj z eSTREAM portfolia `http://www.ecrypt.eu.org/stream/`. Zdrojové kódy jsou v jazyce C a nebyly nějak modifikovány.

Výkon jednotlivých šifer byl měřen v několika kategoriích a to proto, že šifry mohou být nasazeny v různých situacích z nichž každá může mít jiné požadavky na účinnost. Byly zvoleny čtyři výkonostní kritéria a to:

1. Šifrování dlouhých proudů dat – Je to pravděpodobně nejdůležitější kritérium v mnoha aplikacích. Je to test ve kterém mají proudové šifry největší výhodu oproti blokovým šifrám. Rychlost šifrování byla měřena pro pakety o délce 4 kB. Čas pro nastavení klíče ani inicializačního vektoru (IV) nebyl do tohoto testu zahrnut.

2. Rychlost paketového šifrování – Blokové šifry jsou lepší volbou pro velmi krátké pakety, ovšem s rostoucí délkou paketů dokáží být proudové šifry efektivnější. Zjistíme tak proudové šifry, jejíž rychlost šifrování není tak ovlivněna délkami paketů a mají tak výhodu v aplikacích, které používají různé délky paketů. Rychlost paketového šifrování byla měřena pro pakety délky 40, 576 a 1500 bytů. Tyto délky byly vybrány jako reprezentativní vzorek pro provoz na internetu.

3. Agilita – Pokud je potřeba šifrovat mnoho proudů paralelně na jednom procesoru, bude výkon záviset nejen na rychlosti kódů, ale i na čase stráveným přepínáním z jedné relace do druhé.

4. Test rychlosti nastavení klíče a inicializačního vektoru (IV) – Je to poslední test, který samostatně měří rychlost nastavení klíče a inicializačního vektoru (IV). Výsledky tohoto testu asi nejméně ovlivňují rychlost šifrování, protože účinnost nastavení inicializačního vektoru (IV) se odráží v rychlosti paketového

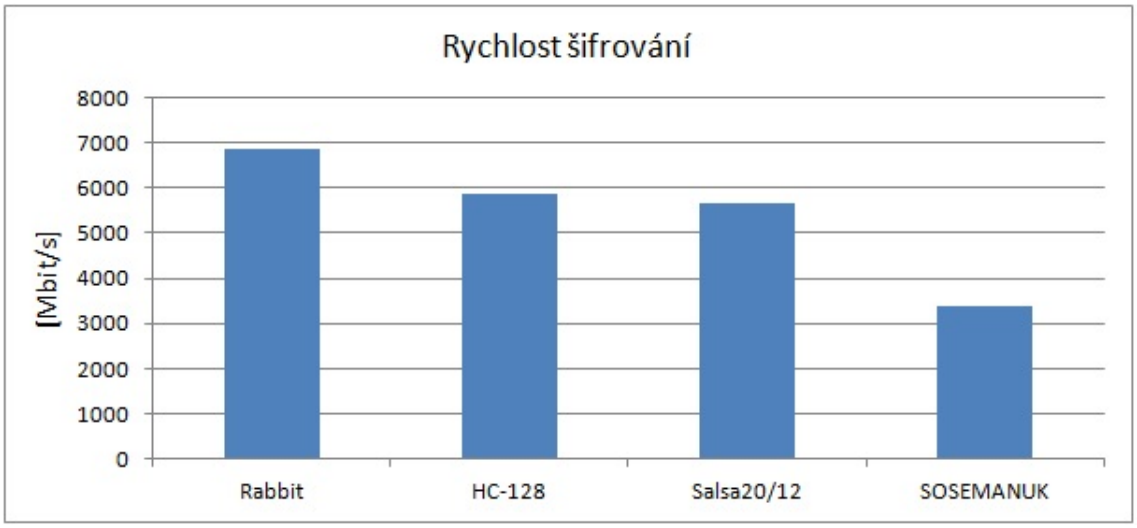
šifrování a nastavení klíče je obvykle zanedbatelné ve srovnání s prací potřebné k výrobě a výměně klíče.

Výkon je většinou uváděný v jednotce cykly/byte, což udává kolik hodinových cyklů procesor bude provádět na jeden byte dat zpracovaných v algoritmu. Tento ukazatel se běžně používá pro reálný výkon kryptografických funkcí.

4.2 Výsledky testování softwarově orientovaných šifer

Tab. 4.1: Tabulka výsledků šifrování dlouhých proudů dat softwarově orientovaných šifer.

Šifra	Délka klíče [bit]	délka IV [bit]	Výkon [cyklů/byte]	Rychlost [Mbit/s]
Rabbit	128	64	2,33	6853,80
HC-128	128	128	2,73	5852,89
Salsa20/12	128	64	2,82	5668,15
SOSEMANUK	256	128	4,71	3392,66



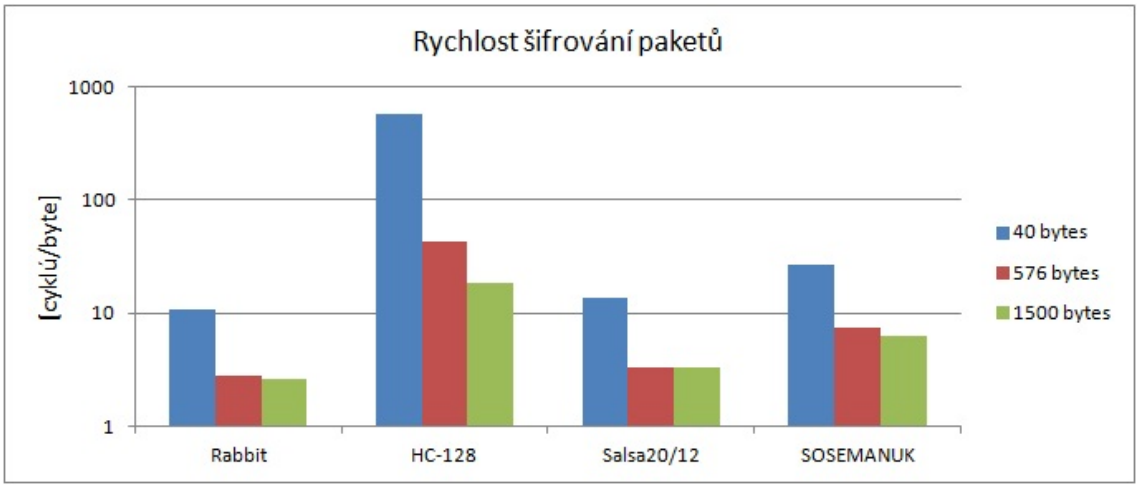
Obr. 4.1: Graf rychlosti šifrování dlouhých proudů dat softwarově orientovaných šifer.

Z měření vyplívá, že pro šifrování dlouhých proudů dat je nejvhodnější šifra Rabbit s rychlostí šífrování 6853,8 Mbit/s, která je více jak dvakrát rychlejší, než poslední šifra SOSEMANUK s rychlostí šifrování 3392,66 Mbit/s. Je to nejspíše způsebeno

dvojnásobnou délkou tajného klíče u šifry SOSEMANUK. Další dva algoritmy HC-128 a Salsa20/12 jsou na tom s rychlostí šifrování dlouhých proudů dat jen o něco hůře, než nejrychlejší šifra Rabbit.

Tab. 4.2: Tabulka výsledků rychlosti paketového šifrování softwarově orientovaných šifer.

Šifra	Délka klíče [bit]	IV [bit]	40 bytes [cyklů/byte]	576 bytes [cyklů/byte]	1500 bytes [cyklů/byte]
Rabbit	128	64	10,90	2,82	2,59
HC-128	128	128	583,32	42,96	18,26
Salsa20/12	128	64	13,45	3,30	3,29
SOSEMANUK	256	128	26,84	7,53	6,34

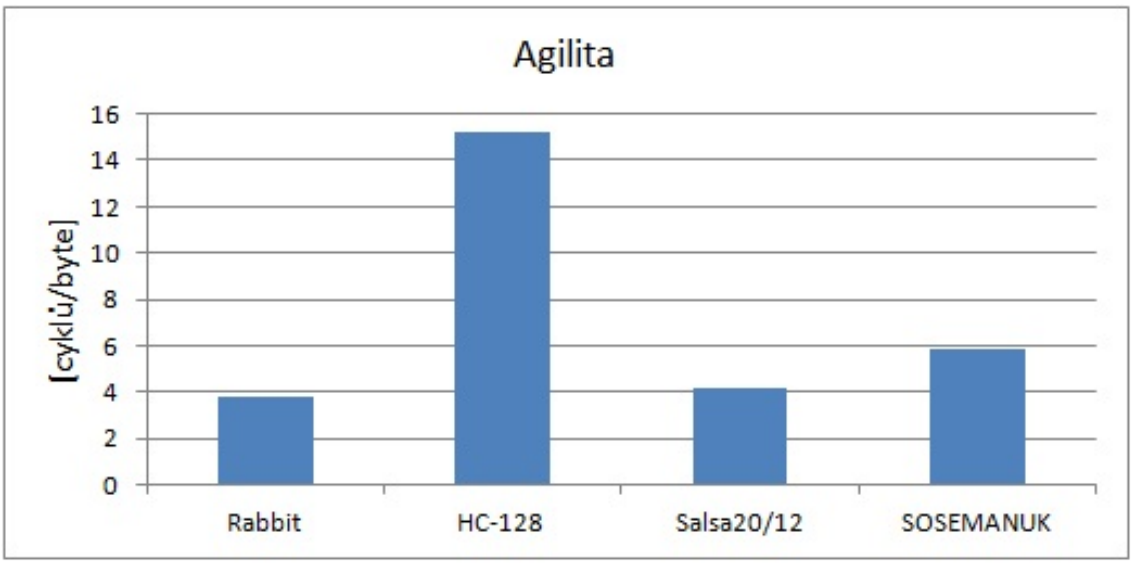


Obr. 4.2: Graf rychlosti šifrování paketů délek 40 bytes, 576 bytes a 1500 bytes softwarově orientovaných šifer.

Z měření rychlosti paketového šifrování vyplynulo, že proudové šifry nejsou vhodné pro šifrování krátkých délek paketů. Při šifrování zprávy délky 40 bytů jsou algoritmy několika násobně pomalejší, než při šifrování dlouhých proudů dat. Z grafu můžeme názorně vyčíst, že čím delší zpráva byla šifrována, tím rychleji šifrování probíhalo. Nejrychlejší byl opět algoritmus Rabbit, jen o něco málo pomalejší byl algoritmus Salsa20/12. Algoritmus SOSEMANUK byl už více jak dvakrát pomalejší, než Rabbit. Nejhůře dopadl algoritmus HC-128 u kterého bylo šifrování všech délek paketů velice pomalé, to je nejspíše způsobeno dlouhou inicializací, tento algoritmus se tedy hodí pouze pro šifrování dlouhých proudů dat.

Tab. 4.3: Tabulka výsledků agility softwarově orientovaných šifer.

Šifra	Délka klíče [bit]	délka IV [bit]	Agilita [cyklů/byte]
Rabbit	128	64	3,84
HC-128	128	128	15,18
Salsa20/12	128	64	4,13
SOSEMANUK	256	128	5,87



Obr. 4.3: Graf výkonu agility softwarově orientovaných šifer.

U měření agility se testuje schopnost šifrovat více proudu dat paralelně na jednom procesoru. Nejlépe si vedl opět algoritmus Rabbit s rychlostí šifrování 3,84 cyklů/byte, jen o něco pomalejší byl algoritmus Salsa20/12 s rychlostí šifrování 4,13 cyklů/byte a algoritmus SOSEMANUK s rychlostí šifrování 5,87 cyklů/byte. Nejpomalejší byl HC-128, který měl rychlost šifrování několika násobně pomalejší a to 15,18 cyklů/byte.

Tab. 4.4: Tabulka výsledků rychlosti nastavení klíče a IV softwarově orientovaných šifer

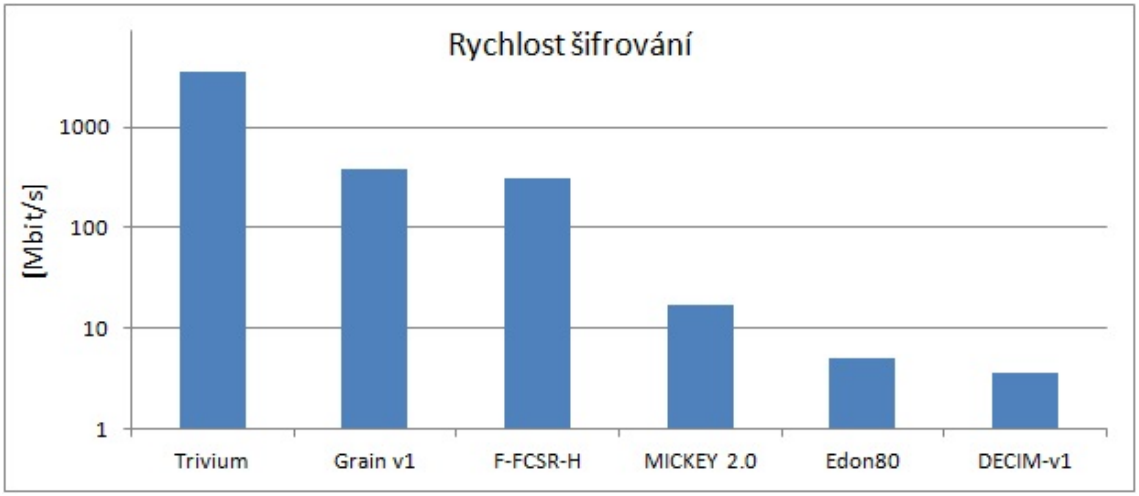
Šifra	Délka klíče [bit]	délka IV [bit]	Rychlost klíče [cyklů/byte]	Rychlost IV [cyklů/bajt]
Rabbit	128	64	243,32	212,59
HC-128	128	128	57,41	23087,14
Salsa20/12	128	64	38,16	22,77
SOSEMANUK	256	128	1038,19	632,20

Tento test nejméně ovlivňuje rychlost šifrování dlouhých proudů dat. Můžeme říci, že rychlost nastavení klíče je téměř zanedbatelná ve srovnání s prací potřebnou k výrobě a výměně klíče. Rychlost nastavení inicializačního vektoru (IV) nejvíce ovlivňuje rychlost šifrování paketů krátkých délek. U algoritmu HC-128 je doba inicializace velmi dlouhá (23087 cyklů/byte), proto šifrování krátkých paketů probíhá pomale ve srovnání s ostatnímy softwarově orientovanými algoritmy.

4.3 Výsledky testování hardwarově orientovaných šifer

Tab. 4.5: Tabulka výsledků šifrování dlouhých proudů dat hardwarově orientovaných šifer.

Šifra	Délka klíče [bit]	délka IV [bit]	Výkon [cyklů/byte]	Rychlost [Mbit/s]
Trivium	80	80	4,43	3611,61
Grain v1	80	64	42,19	379,28
F-FCSR-H	80	80	51,51	310,64
MICKEY 2.0	80	80	936,98	17,09
Edon80	80	64	3179,19	5,03
DECIM-v1	80	64	4470,99	3,58

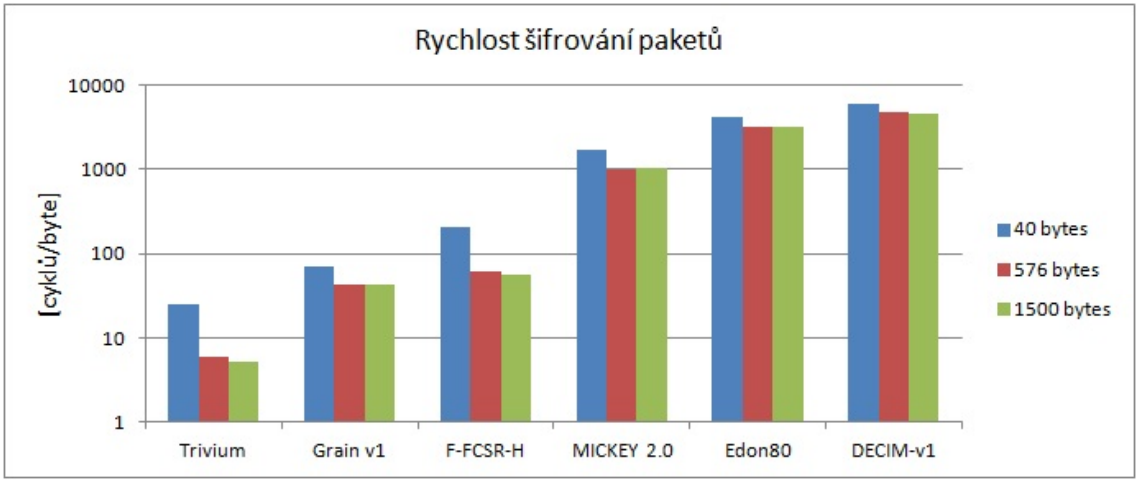


Obr. 4.4: Graf rychlosti šifrování dlouhých proudů dat hardwarově orientovaných šifer.

Z testů šifrování dlouhých proudů dat vyplývá, že nejrychlejší byl algoritmus Trivium s rychlostí šifrování 3611,61 Mbit/s, jak můžeme vyčíst z logaritmického grafu je asi desetkrát rychlejší než algoritmy Grain v1 (379,28 Mbit/s) a F-FCSR-H (310,64 Mbit/s) a asi tisíckrát rychlejší, než algoritmy Edon80 (5,03 Mbit/s) a Decim-v1 (3,58 Mbit/s). Přestože šifra Trivium necílí na softwarové použití, je díky snadné paralelizaci velice výkoná i standardních PC. Šifry Grain v1 a F-FCSR-H mají také poměrně slušnou rychlost šifrování v porovnání se zbývajícímy algoritmy.

Tab. 4.6: Tabulka výsledků rychlosti paketového šifrování hardwarově orientovaných šifer.

Šifra	Délka klíče [bit]	IV [bit]	40 bytes [cyklů/byte]	576 bytes [cyklů/byte]	1500 bytes [cyklů/byte]
Trivium	80	80	24,74	5,89	5,07
Grain v1	80	64	69,58	43,05	41,97
F-FCSR-H	80	80	210,23	62,63	55,81
MICKEY 2.0	80	80	1711,46	977,78	1021,40
Edon80	80	64	4098,17	3237,43	3205,29
DECIM-v1	80	64	6098,00	4700,05	4651,93

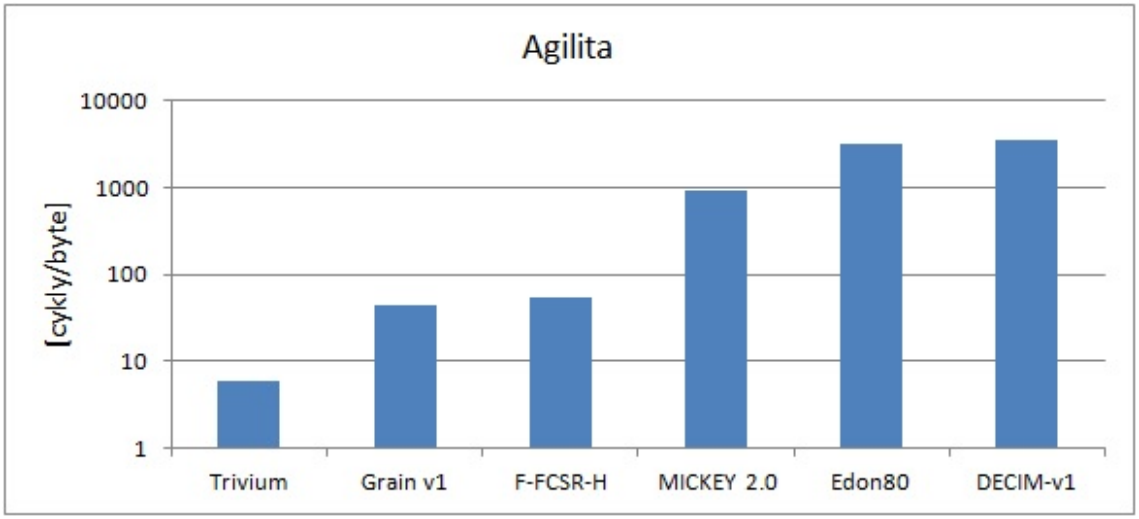


Obr. 4.5: Graf rychlosti šifrování paketů délek 40 bytes, 576 bytes a 1500 bytes hardwarově orientovaných šifer.

Z měření rychlosti paketového šifrování opět vyplynulo, že proudové šifry nejsou vhodné pro šifrování krátkých délek paketů. Nejlépe si vedla šifra Trivium, která zprávu délky 40 bytů šifrovala rychleji (24,74 cyklů/byte), než všechny ostatní algoritmy dokázaly šifrovat dlouhé proudy dat v předešlém testu. Asi třikrát pomalejší v šifrování zprávy 40 bytů byla šifra Grain v1 (69,58 cyklů/byte), následovaná šifrou F-FCSR-H (210,23 cyklů/byte), všechny ostatní pak byly v šifrování řádově pomalejší.

Tab. 4.7: Tabulka výsledků agility hardwarově orientovaných šifer.

Šifra	Délka klíče [bit]	délka IV [bit]	Agilita [cyklů/byte]
Trivium	80	80	6,07
Grain v1	80	64	44,39
F-FCSR-H	80	80	55,06
MICKEY 2.0	80	80	939,73
Edon80	80	64	3188,20
DECIM-v1	80	64	3433,63



Obr. 4.6: Graf výkonu agility hardwarově orientovaných šifer.

U testování agility se ověřuje rychlost šifrování více proudu dat paralelně na jednom procesoru. Nejlépe si vedl opět Trivium, jehož rychlost šifrování byla 6,07 cyklů/byte. Další dva algoritmy byly v rychlosti šifrování poměrně vyrovnaní, Grain v1 (44,39 cyklů/byte) a F-FCSR-H (55,06 cyklů/byte). Ostatní testované algoritmy byly mnohonásobně pomalejší.

Tab. 4.8: Tabulka výsledků rychlosti nastavení klíče a IV hardwarově orientovaných šifer.

Šifra	Délka klíče [bit]	délka IV [bit]	Rychlost klíče [cyklů/byte]	Rychlost IV [cyklů/byte]
Trivium	80	80	40,22	729,52
Grain v1	80	64	14,92	886,70
F-FCSR-H	80	80	30,95	7531,83
MICKEY 2.0	80	80	26,88	30480,00
Edon80	80	64	1811,63	36919,23
DECIM-v1	80	64	14,66	60420,00

Jak již bylo řečeno rychlost nastavení klíče a rychlost nastavení inicializačního vektoru (IV), téměř neovlivňují rychlost šifrování dlouhých proudů dat. Rychlost nastavení klíče je prakticky zanedbatelná ve srovnání s prací potřebnou k výrobě a výměně klíče. Rychlost nastavení inicializačního vektoru (IV) nejvíce ovlivňuje rychlost šifrování paketů krátkých délek. Znamená to, že čím je rychlost nastavení delší, tím pomalejší je šifrování.

4.4 Zhodnocení proudových šifer

Zhodnocení softwarově orientovaných šifer: Nejrychlejší šifrou byl ve všech měřených testech Rabbit, který v nejdůležitějším kritériu, tedy testování dlouhých proudů dat měl propustnost 6853 Mbit/s, jen o něco pomalejší ve všech testech byla šifra Salsa20/12, která v testování dlouhých proudů dat měla propustnost 5668 Mbit/s. Šifra SOSEMANUK byla ve všech testech více jak dvakrát pomalejší, než šifra Rabbit, v testování dlouhých proudů dat dosáhla propustnosti 3392 Mbit/s. Šifru HC-128 lze doporučit pouze k šifrování dlouhých proudů dat, kde měla propustnost 5852 Mbit/s, v ostatních testech byla vždy několika násobně pomalejší, než ostatní testované šifry. Celkově lze říci, že všechny testované softwarově orientované šifry měli velmi dobrou propustnost, ovšem nelze je doporučit k šifrování velmi krátkých zpráv, kde je jejich rychlost šifrování velice pomalá.

Zhodnocení hardwarově orientovaných šifer: Z testovaných hardwarově orientovaných šifer byl nejrychlejší algoritmus Trivium a to ve všech měřených testech. V nejdůležitějším testu šifrování dlouhých proudů dat byla jeho propustnost 3611 Mbit/s. Výrazně pomalejší v šifrování dlouhých proudů dat byla šifra Grain v1 s propustností 379 Mbit/s a šifra F-FCSR-H s propustností 310 Mbit/s. Zbývající šifry, tedy MICKEY 2.0, Edon80 a Decim v1 byly ve všech měřených testech výrazně pomalejší, než šifra Trivium a nelze je tak kvůli nízkému výkonu doporučit k použití na běžných PC. K použití nelze doporučit ani šifru F-FCSR-F u které byla objevena bezpečnostní mezera.

Celkové zhodnocení proudových šifer: Jak se dalo předpokládat mají softwarově orientované šifry daleko větší propustnost, než hardwarově orientované šifry určené pro zařízení s nižším výpočetním výkonem. Ze softwarově orientovaných lze doporučit k použití šifru Rabbit, která byla nejvýkonější ve všech testech, ale i šifru Salsa20/12 a HC-128. Z hardwarově orientovaných lze doporučit šifru Trivium, která byla dokonce vykonější, než softwarově orientovaná šifra SOSEMANUK. Z testů vyplynulo, že proudové šifry nejsou vhodné k šifrování krátkých paketů, ale vhodné k šifrování dlouhých proudů dat.

5 ZÁVĚR

V úvodu bakalářské práce jsem stručně shrnul historii kryptografie a popsal metody zabezpečení zpráv v dávných dobách. Vysvětlil jsem princip a základní pojmy se kterými se v kryptografii setkáme. Popsány byby oblasti symetrického a asymetrického šifrování. V praxi se využívá obou druhů šifrování, přitom je vhodné symetrickou a asymetrickou kryptografii kombinovat, např. symetrický klíč je vhodné šifrovaně přenášet pomocí asymetrické kryptografie a následně pak komunikaci šifrovat pomocí symetrické kryptografie.

V druhé kapitole je prostor věnován lehké kryptografii. Porovnal jsem a zhodnotil nově vzniklé algoritmy jak pro blokové a proudové šifry tak pro hashovací funkce určené pro lehkou kryptografii. Nejvýraznějšími blokovými šiframi pro lehkou kryptografii jsou LBLOCK a PRESENT. Proudovými šiframi pro lehkou kryptografii jsou např. Trivium, Grain v1, MICKEY 2.0 a další. Pro hash funkce byl vyvinut algoritmus PHOTON.

Další kapitola se zabývá implementací algoritmů. Dnes je kladen důraz na správnou implementaci daného algoritmu. Při špatně provedené implementaci můžeme využít mnoha druhů útoků např. posranními kanály, které bývají velmi účinné. Další podkapitola je zaměřena na popis hardwarové i softwarové implementace. Hardwarová implementace je výhodná pro svoji rychlost a bezpečnost, softwarová implementace má výhodu ve flexibilitě, přenositelnosti a snadnosti použití. Jako poslední jsem popsal hardwarová zařízení s omezeným výpočetním výkonem, mezi něž patří smart karty, RFID čipy a mikrokontroléry. Vlastně kvůli nim vznikla lehká kryptografie, která umožní implementaci kryptografických primitiv do těchto zařízení.

V praktické části práce jsem se zaměřil na testování vybraných proudových šifer zveřejněných v projektu eSTREAM a to jak na softwarově orientované šifry tak hardwarově orientované šifry určené pro zařízení s nižším výpočetním výkonem. Testování proběhlo podle čtyř rozdílných kritérií. Ze softwarově orientovaných šifer byl nejvýkonější ve všech testech algoritmus Rabbit. Z hardwarově orientovaných šifer byl nejvýkonější algoritmus Trivium a to ve všech měřených testech. Celkově lze pro použití na běžném PC doporučit softwarově orientované šifry Rabbit, Salsa20/12 a HC-128 a z hardwarově orientovaných šifer algoritmus Trivium. Z testů také vyplívá vlastnost proudových šifer a to, že nejsou vhodné k šifrování paketů krátkých délek. Výsledky všech testů jsou k dispozici na přiloženém CD.

V budoucnu by se další práce mohly zaměřit například na implementaci těchto proudových šifer do hardwarově omezených zařízeních uvedených v kapitole 3.3 a na testování blokových šifer a hash funkcí popsaných v kapitole 2.1 a 2.2.

LITERATURA

[1] AMIROVÁ, K. *Úvod do kryptografie*[online]. 2007 [cit. 5. 12. 2012]. Dostupné z URL: <http://sifrovani.fd.cvut.cz/vyuz_zhod.html>.

[2] BABBAGE, S a M DODD. *The stream cipher MICKEY 2.0*[online]. 2006 [cit. 5. 12. 2012]. Dostupné z URL: <http://www.ecrypt.eu.org/stream/p3ciphers/mickey/mickey_p3.pdf>.

[3] BAYER, T. *Návrh hardwarového šifrovacího modulu*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 62 s. Vedoucí diplomové práce Ing. Jiří Sobotka.

[4] BERBAIN, C et al. *Decim, a new stream cipher for hardware applications*[online]. [cit. 10. 5. 2013]. Dostupné z URL: <http://www.ecrypt.eu.org/stream/ciphers/decim/decim.pdf>.

[5] BERBAIN, C et al. *Sosemanuk, a fast software-oriented stream cipher*[online]. [cit. 5. 5. 2013]. Dostupné z URL: <http://www.ecrypt.eu.org/stream/p3ciphers/sosemanuk/sosemanuk_p3.pdf>.

[6] BERGER, Thierry, Francois ARNAULT a Cédric LAURADOUX. *F-FCSR stream ciphers*[online]. [cit. 10. 5. 2013]. Dostupné z URL: <http://perso.citi.insa-lyon.fr/claurado/publis/ffcsr.pdf>.

[7] BERNSTEIN D. *Salsa20 design*[online]. [cit. 5. 5. 2013]. Dostupné z URL: <http://cr.yp.to/snuffle/design.pdf>.

[8] BOESGAARD, M at al. *The Stream Cipher Rabbit*[online]. [cit. 5. 5. 2013]. Dostupné z URL: <http://www.ecrypt.eu.org/stream/p3ciphers/rabbit/rabbit_p3.pdf>.

[9] CANNIÉRE, C a B PRENEEL. *Trivium Specifications*[online]. [cit. 10. 5. 2013]. Dostupné z URL: <http://www.ecrypt.eu.org/stream/p3ciphers/trivium/trivium_p3.pdf>.

[10] CHYTIL, V. *Bezpečnost dat v informatice*. Zlín: Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, 2011. 77 s. Vedoucí bakalářské práce RNDr. Ing. Miloš Křemář.

[11] FORMAN, T. *Portál pro podporu výuky kryptografie*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 96 s. Vedoucí diplomové práce doc. Ing. Václav Zeman, Ph.D.

[12] GANIAN, R. *Testování rychlosti proudových šifer*. Brno: Masarykova univerzita, Fakulta informatiky, 2006. 23 s. Vedoucí bakalářské práce Mgr. Jan Krhovják.

[13] GEYER, L. *Eliptické křivky v kryptografi*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 34 s. Vedoucí bakalářské práce Ing. Petra Lambertová.

[14] GLIGOROSKI, D at al. *A complete description of Edon80*[online]. [cit. 10. 5. 2013]. Dostupné z URL: <`http://www.ecrypt.eu.org/stream/p3ciphers/edon80/edon80_p3.zip`>.

[15] HAMPL, D. *Kryptografie na výpočetně omezených zařízeních*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 50 s. Vedoucí bakalářské práce Ing. Lukáš Malina.

[16] HELL, Martin, Thomas JOHANSSON a Willi MEIER. *Grain - A Stream Cipher for Constrained Environments*[online]. [cit. 10. 5. 2013]. Dostupné z URL: <`http://www.ecrypt.eu.org/stream/p3ciphers/grain/Grain_p3.pdf`>.

[17] HONGJUN, W. *The Stream Cipher HC-128*[online]. [cit. 5. 5. 2013]. Dostupné z URL: <`http://www.ecrypt.eu.org/stream/p3ciphers/hc/hc128_p3.pdf`>.

[18] KARGER, M. *WWW průvodce moderní kryptografií*. Zlín: Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, 2008. 64 s. Vedoucí bakalářské práce Ing. Karel Perůtka, Ph.D.

[19] KLÍMA, V. *Co to je lehká kryptografie*[online]. 2011 [cit. 10. 5. 2013]. Dostupné z URL: <`http://cryptography.hyperlink.cz/2011/ST_2011_10_16_17.pdf`>.

[20] KLÍMA, V. *Hash rychlejší než foton*[online]. 2011 [cit. 5. 5. 2013]. Dostupné z URL: <`http://cryptography.hyperlink.cz/2011/ST_2011_11_16_16.pdf`>.

[21] KŘÍŽ, J. *Softwarová podpora výuky kryptosystémů založených na problému diskrétního logaritmu*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 74 s. Vedoucí diplomové práce doc. Ing. Karel Burda, CSc.

[22] LORENC, V a V MATYÁŠ. *Autentizační HW a možná vylepšení*[online]. 2007 [cit. 7. 12. 2012]. Dostupné z URL: <`http://www.ics.muni.cz/bulletin/articles/563.html`>.

[23] MENEZES, Alfred J, Paul C VAN OORSCHOT a Scott A VANSTONE. *Handbook of applied Cryptography*. USA: CRC Press, 1996. 816 s. ISBN 0-8493-8523-7.

[24] MILOŠ, J. *Kryptografické metody zabezpečení dat* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 43 s. Vedoucí bakalářské práce Ing. Petra Lambertová.

[25] MOLLIN, Richard A. *An Introduction to Cryptography*. USA: Network Associates, 1998. ISBN 15-848-8618-8.

[26] OCHODKOVÁ, E. *Přínos teorie eliptických křivek k řešení moderních kryptografických systémů*[online]. 2003 [cit. 26. 11. 2012]. Dostupné z URL: <`http://www.cs.vsb.cz/arg/workshop/files/ecc_eli.pdf`>.

[27] PIPER, F a Sean MURPHY. *Kryptografie*. 1. vyd. v českém jazyce. Překlad Pavel Mondschein. Praha: Dokořán, 2006. 157 s. ISBN 80-736-3074-5.

[28] POP, T. *Kryptografie a její použití při zabezpečeném přenosu datových souborů*. Praha: Univerzita Karlova v Praze, Fakulta Matematicko-fyzikální, 2006. 52 s. Vedoucí bakalářské práce Mgr. Drahomíra Doležalová-Spoustová.

[29] POPOVSKÝ, M. *Útoky postrannímí kanály*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 71 s. Vedoucí diplomové práce Ing. Zdeněk Martinásek.

[30] POSCHMANN, Axel Y. *LIGHTWEIGHT CRYPTOGRAPHY: Cryptographic Engineering for a Pervasive World*. Bochum: Ruhr-University Bochum, Faculty of Electrical Engineering and Information Technology, 2009. 197 s.

[31] POSPÍŠIL, K. *Výkonnostní testy kryptografických algoritmů*. Brno: Vysoké učenítechnické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 54 s. Vedoucí bakalářské práce Ing. Jiří Sobotka

[32] SCHNEIER, B. *Applied cryptography: protocols, algorithms, and source code in C*. New York: Wiley, 1996. 758 s. ISBN 04-711-1709-9.

[33] SINGH, S. *Kniha kódů a šifer: tajná komunikace od starého Egypta po kvantovou kryptografii*. Praha: Dokořán, 2003. 382 s. ISBN 80-865-6918-7.

[34] STEINER, O. *Svět šifer aneb lehký úvod do kryptografie*[online]. 1999, poslední aktualizace 2001 [cit. 10. 11 2012]. Dostupné z URL: <`http://web.iol.cz/steiner/kryptografie/index.html`>. ‘

[35] VONDRUŠKA, Pavel. *Crypto-World*[online]. 2012 [cit. 10. 5. 2013]. Dostupné z URL: <http://crypto-world.info/casop14/crypto02_12.pdf>.

[36] WALEK, V. *Moderní asymetrické kryptosystémy*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. 63 s. Vedoucí diplomové práce Ing. Lukáš Malina.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

3DES (Triple Data Encryption Standard)

DES (Data Encryption Standard)

DH (Diffie–Hellman)

DMA (Direct Memory Access)

DSA (Digital Signature Algorythm)

EAS (Advanced Encryption Standard)

ECC (Eliptic Curve Cryptography)

EEPROM (Electrically Erasable Programmable Read-Only Memory)

EPROM (Erasable Programmable Read-Only Memory)

ETHZ (Swiss Federal Institute of Technology)

GE (Gate Equivalent)

GNU (General Public License)

GPG (GNU Privacy Guard)

GSM (Groupe Spécial Mobile)

HTTPS (Secure Hypertext Transfer Protocol)

IDEA (International data encryption algorithm)

IO (Input Output)

IV (Initialisation vector)

MD5 (Message Digest 5)

MICKEY (Mutual Irregular Clocking KEYstream generator)

NIST (National Institute of Standards and Technology)

PGP (Pretty Good Privacy)

RC4 (Rivest Cipher 4)

ROM (Radio Frequency Identification)

RFID (Read-Only Memory)

RSA (Rivest, Shamir, Adleman)

SHA (Secure Hash Algorithm)

SQL (Structured Query Language)

SSL (Secure Sockets Layer)

WEP (Wired equivalent privacy)

WPA (Wi-Fi protected access)

XOR (Exclusive Or)